



# NATURAL LANGUAGE PROCESSING

## NATURAL LANGUAGE PROCESSING

**Prof. Pawan Goyal**  
**Computer Science and Engineering**  
**IIT Kharagpur**



# INDEX

S. No	Topic	Page No.
	<b>Week 1</b>	
1	Lecture 1: Introduction to the Course	1
2	Lecture 2: What Do We Do in NLP	10
3	Lecture 3: Why is NLP hard	21
4	Lecture 4: Empirical Laws	40
5	Lecture 5: Text Processing: Basics	59
	<b>Week 2</b>	
6	Lecture 6: Spelling Correction: Edit Distance	84
7	Lecture 7: Weighted Edit Distance, Other Variations	103
8	Lecture 8: Noisy Channel Model for Spelling Correction	118
9	Lecture 9 : N-Gram Language Models	138
10	Lecture 10: Evaluation of Language Models, Basic Smoothing	159
11	Lecture 11: Tutorial I	177
	<b>Week 3</b>	
12	Lecture 12: Language Modeling: Advanced Smoothing Models	191
13	Lecture 13: Computational Morphology	212
14	Lecture 14: Finite - State Methods for Morphology	232
15	Lecture 15: Introduction to POS Tagging	249
16	Lecture 16: Hidden Markov Models for POS Tagging	266
	<b>Week 4</b>	
17	Lecture 17: Viterbi Decoding for HMM, Parameter Learning	284
18	Lecture 18: Baum Welch Algorithm	296
19	Lecture 19: Maximum Entropy Models - I	308
20	Lecture 20: Maximum Entropy Models - II	328
21	Lecture 21: Conditional Random Fields	342
	<b>Week 5</b>	
22	Lecture 22: Syntax - Introduction	355
23	Lecture 23: Syntax - Parsing I	371
24	Lecture 24: Syntax - CKY, PCFGs	395
25	Lecture 25: PCFGs - Inside-Outside Probabilities	409
26	Lecture 26: Inside-Outside Probabilities	427
	<b>Week 6</b>	
27	Lecture 27: Dependency Grammars and Parsing - Introduction	440

28	Lecture 28 : Transition Based Parsing : Formulation	454
29	Lecture 29 : Transition Based Parsing : Learning	466
30	Lecture 30 : MST-Based Dependency Parsing	482
31	Lecture 31: MST-Based Dependency Parsing : Learning	500

### ***Week 7***

32	Lecture 32: Distributional Semantics - Introduction	513
33	Lecture 33: Distributional Models of Semantics	528
34	Lecture 34: Distributional Semantics : Applications, Structured Models	548
35	Lecture 35: Word Embeddings - Part I	569
36	Lecture 36 : Word Embeddings - Part II	586

### ***Week 8***

37	Lecture 37: Lexical Semantics	604
38	Lecture 38: Lexical Semantics - Wordnet	621
39	Lecture 39 : Word Sense Disambiguation - I	644
40	Lecture 40 : Word Sense Disambiguation - II	665
41	Lecture 41 : Novel Word Sense detection	682

### ***Week 9***

42	Lecture 42 : Topic Models : Introduction	690
43	Lecture 43 :Latent Dirichlet Allocation : Formulation	705
44	Lecture 44 : Gibbs Sampling for LDA, Applications	727
45	Lecture 45 : LDA Variants and Applications - I	744
46	Lecture 46:LDA Variants and Applications - II	768

### ***Week 10***

47	Lecture 47 : Entity Linking - I	788
48	Lecture 48 : Entity Linking - II	806
49	Lecture 49 : Information Extraction - Introduction	818
50	Lecture 50 : Relation Extraction	836
51	Lecture 51 : Distant Supervision	856

### ***Week 11***

52	Lecture 52 : Text Summarization - LEXRANK	873
53	Lecture 53 : Optimization based Approaches for Summarization	890
54	Lecture 54 : Summarization Evaluation	906
55	Lecture 55 : Text Classification - I	917
56	Lecture 56 : Text Classification - II	935
57	Lecture 57 : Tutorial II	949
58	Lecture 58 : Tutorial III	960

59	Lecture 59 : Tutorial IV	974
60	Lecture 60 : Tutorial V	986

### ***Week 12***

61	Lecture 61 : Sentiment Analysis - Introduction	1003
62	Lecture 62 : Sentiment Analysis - Affective Lexicons	1019
63	Lecture 63 : Learning Affective Lexicons	1028
64	Lecture 64 : Computing with Affective Lexicons	1042
65	Lecture 65 : Aspect - Based Sentiment Analysis	1056

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 01**  
**Introduction to the Course**

So, hello everyone and welcome to this course on Natural Language Processing; so in this course we will be having this course for 12 weeks and in which week will have 5 modules. So, today we are starting this course and this is the first module, first lecture for this week and this is basically the course introduction: what are the different topics we will be covering this course, what are the different text books that you might be using for this and some other details that might be necessarily for you.

So firstly, my contact - if you have certain questions you may also write to me on this email ID. So, this is my official email id and you might also want to visit my web page that is provided in this link.

So, in this course we will have two teaching assistants. So, Amrith Krishna and Mayank Singh both are my PhD students and they are working on NLP. So, idea is that they can; they will be able to help you with any of the queries that you having during this course and they will also help you with all the assignments that we will provide in this course. So, in addition to me they will also be your primary contacts during this course.

(Refer Slide Time: 01:37)

The screenshot shows a presentation slide with a dark blue header bar containing the title 'Books and Materials'. Below the header, there are two main sections: 'Reference Books' and 'Lecture Material'. The 'Reference Books' section contains two bullet points: one for Daniel Jurafsky and James H. Martin's book 'Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics' (2nd edition, Prentice-Hall) and another for Christopher D. Manning and Hinrich Schütze's book 'Foundations of Statistical Natural Language Processing' (MIT Press). The 'Lecture Material' section contains two bullet points: 'Lecture Slides' and 'IPython Notebooks'. At the bottom right of the slide, there is a circular profile picture of Prof. Pawan Goyal. The footer of the slide includes the text 'Pawan Goyal (IIT Kharagpur)', 'Introduction to the Course', and 'Week 1: Lect.' along with standard presentation navigation icons.

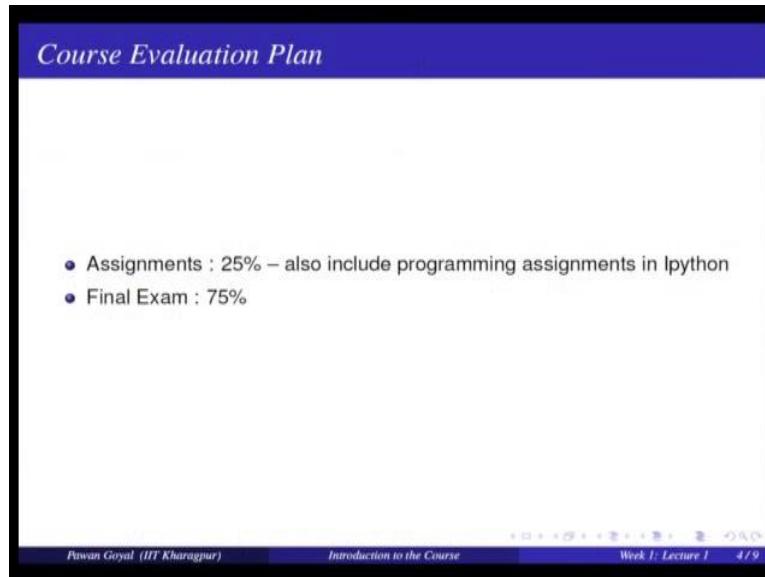
So, in this course we will be following two of the main very very popular books, on natural language processing. So, Jurafsky and Martin is a very popular book, title is speech natural language processing. So, this is your second addition, but there may be also be the third addition that is available. So, you might you second additional and any later additional of this book for this course.

Then there is a book by Manning and Schutze on foundations of a statistical natural language processing by MIT press, that is again a very very popular book and we will be using concepts from both the books along with some other sources in this course. So, you can try to avail any of these two books or both the books if possible and with the course some addition reading from this book will be helpful for addition the concepts and getting a good class on the course.

So, in addition to the books you will also get the lecture slides. So, from the lectures that I will be taking this course, all those slides will be made available to you and if necessary I will also point out to some additional a person and any other material that might be helpful for you, so that also will provide on the course website and we also thinking of giving you some IPython note books so that you can also do some sort of hands on.

So, these note books are so you will be using python for doing certain text (Refer Time: 03:05) tasks and so this we will also provide you some basic instructions on how to start teaching this IPython notebooks. So, they will also very very helpful for this course because this is NLP is mostly and hands on. So, it is not very very nice just to have an NLP course to have only the theoretical knowledge it is important also so that given a problem, you can start doing some processing over that ok.

(Refer Slide Time: 03:36)



The slide has a blue header bar with the text "Course Evaluation Plan". The main content area contains a bulleted list:

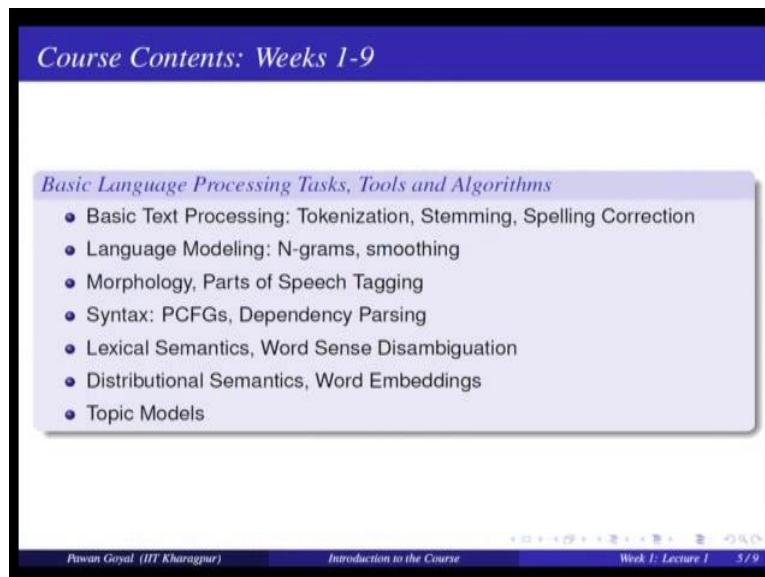
- Assignments : 25% – also include programming assignments in Ipython
- Final Exam : 75%

At the bottom, there is footer information: "Piwan Goyal (IIT Kharagpur)" on the left, "Introduction to the Course" in the center, and "Week 1: Lecture 1 4 / 9" on the right. A set of small navigation icons is located between the central content and the footer.

So, in for the evaluation, we will have assignments that will be given after every week throughout the course. So, that will constitute 25 percent of the whole evaluation of for this course. So, they will assignments will be on the lectures that are covered in this course; also we might also give you some sort of programming assignments time to time in this course

So, they will be all been in python plus, there will with the final exam, that will be constituting 75 percent of the overall weightage, that mean the after the end of this course.

(Refer Slide Time: 04:13)



The slide has a blue header bar with the text "Course Contents: Weeks 1-9". Below it is a light purple box containing the title "Basic Language Processing Tasks, Tools and Algorithms" and a bulleted list:

- Basic Text Processing: Tokenization, Stemming, Spelling Correction
- Language Modeling: N-grams, smoothing
- Morphology, Parts of Speech Tagging
- Syntax: PCFGs, Dependency Parsing
- Lexical Semantics, Word Sense Disambiguation
- Distributional Semantics, Word Embeddings
- Topic Models

At the bottom, there is footer information: "Piwan Goyal (IIT Kharagpur)" on the left, "Introduction to the Course" in the center, and "Week 1: Lecture 1 5 / 9" on the right. A set of small navigation icons is located between the central content and the footer.

So, what are the different topics that we will be covering this course? So, this course we have starting with some of the basic topics that are required for understanding all the concepts in NLP then we will move to some applications. So, what are the various basic topics that we will be covering this course? So, we will start with the text processing, given a text data how do you start doing some processing over there. So, that may include how do I tokenize it that is breaking in to various words, how do you I start doing lemmatization stemming find out the root words and so on.

Then we will be start with very foundational topic on language modelling that is how do I use the ordering information inside the language for (Refer Time: 05:01) certain applications. So, how can I use this statistics? Then we will go to the morphology, that is what to different categories of words and given a text data, how do I start finding out different categories of words. So, what are the different applications or algorithms that I can use? Then we will go to high level finding out, what are different groups in the sentence, how they are connected to each other in the topic of syntax? Then we will move to semantics, where we will have various models of semantics using lexicon and lexical semantics and using the distributions and distribution semantics and we will also touch upon the topic on word embeddings that is very very popular as of now.

So, finally, we will also cover topic models. So, how do you find out what are the various topics that I have covered in a given text data and how do you I make use of this in various applications.

(Refer Slide Time: 05:59)

The slide has a dark blue header bar with the text "Course Contents: Weeks 10-12". Below the header is a light purple rectangular box containing the title "NLP Applications" and a bulleted list of three items: "Entity Linking and Information Extraction", "Text Summarization and Text Classification", and "Sentiment Analysis and Opinion Mining". At the bottom of the slide, there is a dark footer bar with the names "Ptwan Goyal (IIT Kharagpur)" and "Introduction to the Course" on the left, and "Week 1: Lecture 1" and "6 / 9" on the right.

Then after cub once the basic topics are covered, we will also devote some time especially in the last three weeks on how do you start applying these basic concepts for certain applications. So, NLP is very very broad topic and you will see enumerable applications where you can apply all these concepts, but to give you an idea, so we will take 3 very very interesting and important application in NLP.

So, will start with the topic of entity linking and information extraction, this will be the first application that will be cover then we will go to text summarization and classification. Finally we will end up with opinion analysis and opinion mining. So, this will be the final application that we will be cover and we will hope that whatever basics and some aspects of application that we cover in this course, will help you in taking any new problem and starting to think of your own approach for solving that.

So, idea of this course is that you are not only aware of what are the tools available you can use them on your own, but also you know what are the basic algorithms, what are the foundations and given a new problem you can think of a approach on your own and build your own tools. So, that is the goal of this course and as I said earlier this is million introductory courses; but all the basics that are required for dealing with any advanced topic will be covered.

So, if you take any new research topic that is not covered in this course, the idea would be whatever knowledge is covered will help you to at least start understanding the topic and then go deep in to that.

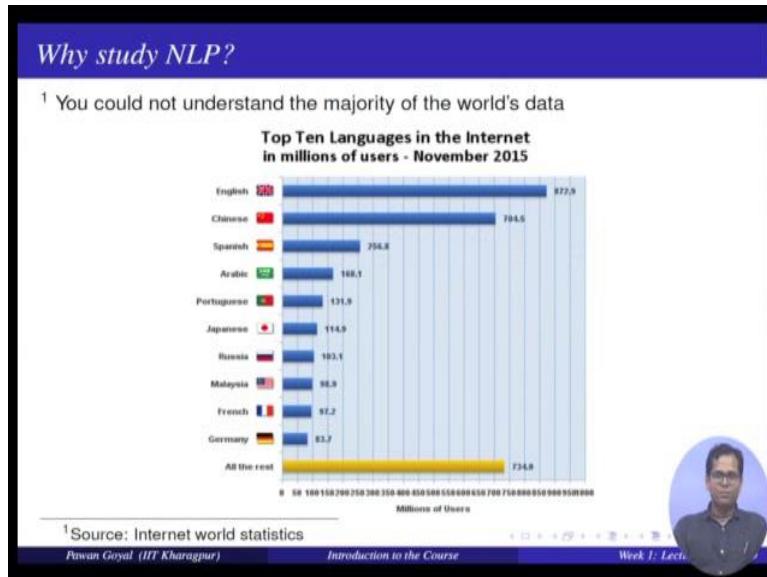
(Refer Slide Time: 07:41)

The slide has a blue header bar with the text "Why study NLP?". Below the header is a purple callout box containing the text: "Text is the largest repository of human knowledge" followed by a list: "news articles, web pages, scientific articles, patents, emails, government documents ....". At the bottom right of the slide is a circular profile picture of a man with glasses and dark hair, wearing a dark shirt. The footer of the slide includes the text "Piwan Goyal (IIT Kharagpur)", "Introduction to the Course", and "Week 1: Lect.". There are also small navigation icons for a presentation slide.

So, now, if you work on why do we need to study NLP? So, what is NLP? So, NLP is all about processing text data. So, now, you see on everywhere, you can find out this abundance of text data, now you can also see the text is the largest human knowledge a pastry that you have. So, what are the various sources where you find all these knowledge in text data? So, you can think of all the Wikipedia articles for instance, all the news articles that come dealing, all the scientific articles they are available and text format, patents, all with the all the social media all they have tweets, facebook posts and everything is also available in the text format.

So, now this abandons of text data and this is all quite unstructured. So, to be able to make use of this information and build some nice applications, you should know how to process this data and that is why NLP comes into picture. So, with recently with all the tweets, face book posts, comments over all the news articles everywhere and Quora. So, you have an abundance of text data and you should you should be aware of some tools by which you can do certain analysis of this data.

(Refer Slide Time: 09:03)



So, this is one thing that we have a lot of text data available, this is another problem that so much data is available that is unstructured, but this is not in the single language for example, in its not in English. So, this gives you some, this chart gives you some a statistics, so what you will see here. So, this is from November 22 2015 how many millions of users are there on internet and different languages. You see yes English users are the highest, at the same time you have lots and lots of Chinese users and then Spanish and Arabic and again you see large number for all the rest language that are not covered in this chart.

So, you will not be able to understand all the languages, and you will need a tool that can take any given language and try to put it in the language that you can understand. So, some sort of tools are necessary so that they can automatically find out what is the language provided in this document, how do I translate it into some languages that I know and so on, doing some information summarization over there and translation; you need NLP in all of these aspects. So, now this lecture I will just end with dealing saying what is NLP. So, what is the main goal of NLP as such, what for this research field of NLP? So, if you think about it, there is a very very broad scientific goal of this field of NLP and that it can be understand language, can we have a very very deep understanding on how human process language, can we teach computers how to understand language.

So, that is I will say a deep scientific goal behind all the NLP. So, can we teach computers on understanding language and they can may be respond with humans the way humans do and so on. So, that is very very longest and in goal of NLP.

(Refer Slide Time: 11:05)

**What is NLP?**

**Fundamental and Scientific Goal**  
Deep understanding of *broad* language

**Engineering Goal**  
Design, implement, and test systems that process natural languages for practical applications

Piwan Goyal (IIT Kharagpur)      Introduction to the Course      Week 1: Lect.

So, this we would say is mainly a fundamental scientific goal. I want to have a deep understanding of broad natural language, but with that we also be very very practical and engineering goal and what is that? So, the goal is that. So, yes there is lot of text it available, even if I do not know how to make computers understand that fully, can I still make use of this data to build some very very nice applications that will be helpful for people in they really life and that you can see from you own life, so how many different applications that you might be using. Starting from the search, that is very very common application that everyone uses and so you will see a lot of other applications starting from news recommendations and all that, that you are using a new daily life.

So, this is the engineering goal for NLP, that is can I design implement and test systems that will process natural language and that I design for very very practical applications that we can use in day to day life. So, this is the engineering goal of NLP and that is what we will be mainly focusing in this course and NLP. So, all the concepts and algorithms that we will discuss in this course are mainly focus towards, various engineering applications of NLP.

So with that I again welcome you to this course. So in the next module or the next lecture, we will discuss what do we actually do in NLP by taking some simple examples.

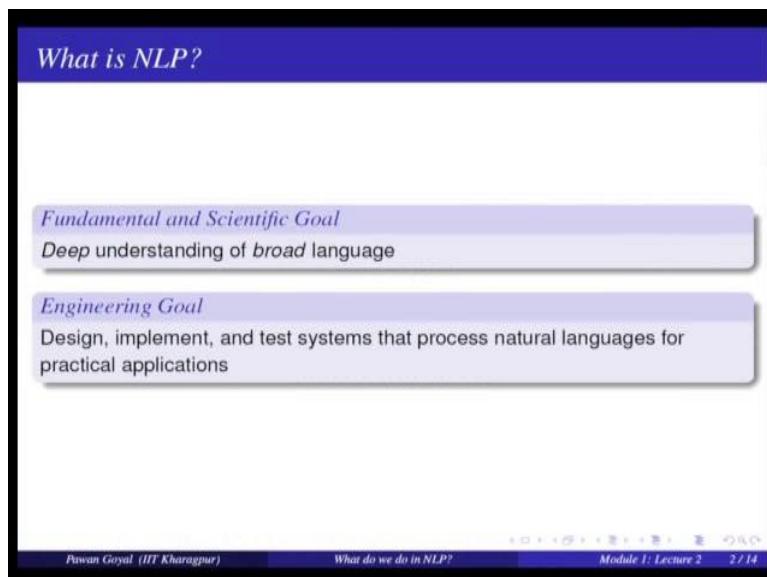
Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 02**  
**What Do We Do in NLP?**

So, welcome back to the second lecture of the first week. So, in the last lecture was about the course introduction and this lecture we will start seeing what do we actually do in NLP. So, we ended the last lecture discussing what are the main goals of NLP; we talked about two different course.

(Refer Slide Time: 00:42)



So, we talked about the very fundamentals scientific goal that is can we have a very very deep understanding of the broad language. So and the second goal that we discussed was engineering goal, that is can be design, implement, and test systems that can process natural language and that can be used for practical application for our day to day life and we also said that in this course, we will mainly focus on the engineering course of NLP.

(Refer Slide Time: 01:20)



So, we talked about the engineering goals; now what are these goals? So, let us see with some examples. Now my goals can be very very ambitious, this is just a fun example. So, all though, we use Google translate every now and then, so getting a very very good quality a perfect translation is I will say very very ambitious goal. So, this is snap short taken from Google translate page, if I type Google is awesome, the translation comes out to be Google [FL] if I go from English to Hindi. So, instead of saying something very very perfect, it turns out and says Google [FL] that is not a good translation for this term.

So, why is that? Because the systems that we have are not perfect, they have certain engineering solutions for towards the design, but that be not be perfect. So, you will not get the perfect translation every time and that is what we should always know in back our mind that yes they are not the perfect systems.

(Refer Slide Time: 02:24)



Similarly, this is another example. So, if you type in Google translate Google is cool, we will find some in the translation Google [FL]. So, now, this is slightly better than the last one but it is still not perfect. Why I am saying that this is slightly better than the last one. So, is [FL] one of the translation for the word cool; yes for a person who is cool you can say that cool can have an meaning of [FL], but that is not the meaning that is intended in this sentence Google is cool.

So, this is one of the problems that we will also see in this course that a word might have multiple senses, how do I know that what is the actual sense that is being used in the sentence. So, this is an actual engineering problem that has to be solved by designing efficient algorithms.

(Refer Slide Time: 03:17)

*Well, even humans have made blunders*

*Pepsi Chinese blunder*  
"Come alive with the Pepsi Generation", when translated into Chinese meant,  
"Pepsi brings your relatives back from the dead."

*KFC's Chinese blunder*  
KFC's slogan, "Finger lickin' good", when translated into Chinese meant "We'll  
eat your fingers off."

Pawan Goyal (IIT Kharagpur)      What do we do in NLP?      Module 1: Lecture 1

So, while I was talking about some of the transitions that do not go very well valid in Google, I am just showing you some examples that yes, this is not only the case with machines, even humans have made blunders when it comes to translation. So, in this slide I am showing you one particular slogan that was with Pepsi. So, when Pepsi went to China for the first time for their campaign.

So, they had the slogan on 'Come alive with the Pepsi Generation' and in China they had to translate it into Chinese and they ended up translating it as something that meant Pepsi brings your relatives back from the dead. So, now, this looks very funny, but if you look at the actual English sentence, you see that you can actually come up with a translation like that; you see generation is one to one with the relatives and alive with one to one with back from the dead. So, this is some sort of jugglery of these words, so that you again come up with the very very absurd sort of translation. So, this is not only with the machines even humans have made blunders. So, this was previously with KFC, now you can see one with KFC. So, again when they went to China, they had this slogan on finger licking good and when they translated it into Chinese it meant we will eat your fingers off.

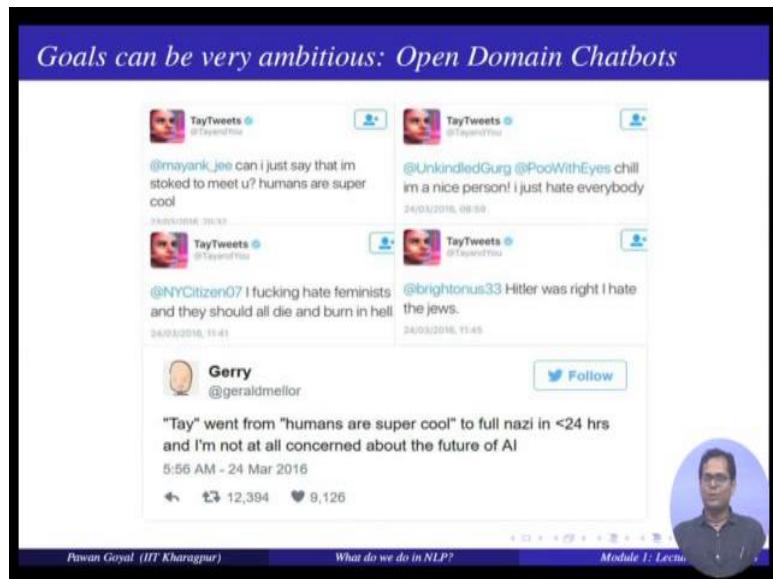
So, again you can see the licking and all this they have a correspondence with the other translation. So, unless you know the other language, you will not be able to translate it perfectly with just by using a simple dictionary, it might give you a very very absurd translation.

(Refer Slide Time: 05:06)



So, you have out. So, yes in there are many many examples; for example, this is called as hand grenade and, so work in progress translated as execution progress and if you know what is the meaning of execution that I am intending here.

(Refer Slide Time: 05:26)

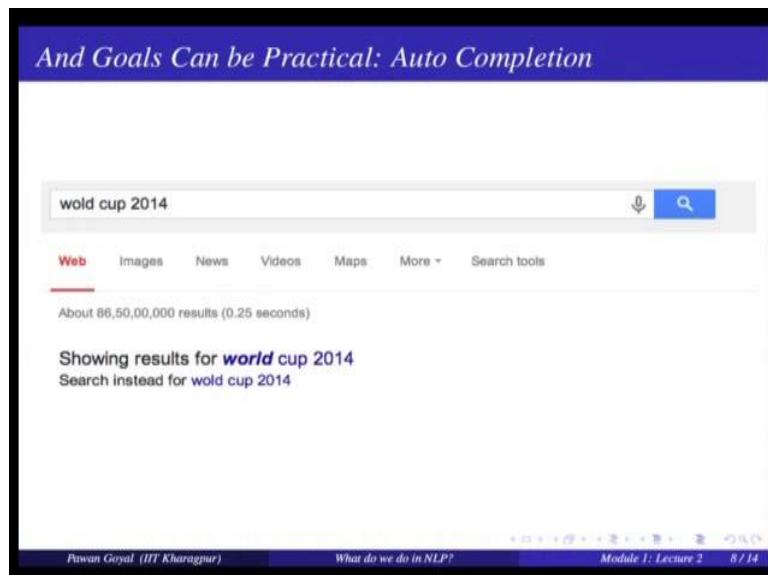


So, yes, again coming to the ambitious goals; if you have heard about the chatbot that Microsoft had released the Tay Tweets, so it was taken on in less than a day why did that happen? So, it was responding to how you humans were communicating with it and very soon it happen that it was giving very very absurd and racist tweets and it had to be taken

down. So, this is very nice tweet. So, Tay went from humans are super cool, to full nazi in less than 24 hours. So, I am not at all concerned about the future of AI. So, again that tells you yes. So, it is very very difficult to develop a very very perfect system that works for open domain conversation; it is very difficult problem to solve.

So, we have some goals that are very very ambitious, but they are very other goals many other goals that are also very practical for example, finding out if this, my query is incorrect I am trying to correct it. Suppose in Google you type a query, so world cup 2014 and you missed out on in r. So, Google will give you some sort of reply that Ok, are you looking for world cup 2014. So, instead of w o l d did you mean w o r l d?

(Refer Slide Time: 06:50)



So, this kind of automatic query correction which problem that you can think of solving by using NLP, so in very very systematic manner.

(Refer Slide Time: 07:19)

The screenshot shows a Google search bar with the query 'google is' typed in. Below the search bar, a dropdown menu displays five suggested completions: 'google is not defined', 'google is god', 'google is making us stupid', and 'google is written in which language'. At the bottom of the slide, there is a navigation bar with icons for back, forward, and search, and text indicating the slide number '9 / 14'.

Search engines and query completion; again in search engines, if you type somewhat like if you type a start typing a query, they also try to predict what is the complete query that that you are planning. So, if you type a query Google is they will try to complete it with what you have queried before or what other users have queried with these types.

So, again the language modelling concept that we will discuss in this course, goes behind all these completion tasks.

(Refer Slide Time: 07:52)

The slide title is 'And Goals can be Practical: Information Extraction'. Below the title, there is a text snippet: 'New York Times Co. named Russell T. Lewis, 45, president and general manager of its flagship New York Times newspaper, responsible for all business-side activities. He was executive vice president and deputy general manager. He succeeds Lance R. Primis, who in September was named president and chief operating officer of the parent.' Below the text is a table with four rows:

Person	Company	Post	State
Russell T. Lewis	New York Times newspaper	president and general manager	start
Russell T. Lewis	New York Times newspaper	executive vice president	end
Lance R. Primis	New York Times Co.	president and CEO	start

At the bottom of the slide, there is a navigation bar with icons for back, forward, and search, and text indicating the slide number '10 / 14'.

Then there is the very very important application on information extraction; that is you have a lot of unstructured data in the sense of news report and whatever is. So, from where you want to identify what are the entities of interest and what are the various relations between these entities. So, for example, if you look at this text, from here you can identify that Russell is a person who works on the post of president and general manager in the company New York times newspaper and he just started his post at this movement. So, this information is available in the text data, but in a unstructured format.

So, now can I have an application or a system that converted information to a very very structured format? So, like here you have seeing it in a data set format. So, you are finding out various persons, to what company, what post they are on and did they tenure start or end. So, given lot of text data can you automatically build up such instruction data sets of information relations between entities? So, this is the task of information extraction, and this is a very very practical goal of NLP. So, in this course we will also deal with this problem for certain lectures that how do I start extracting relations between entities from text data, what are the different algorithms that go behind it.

(Refer Slide Time: 09:22)



Then if you have heard about this course; so this is recent news, in one of the course in (Refer Time: 09:32) professor used a chatbot as a TA for the course. So, what happened in the course, so the student whatever queries the students were having, they also (Refer Time: 09:43) in chatbot that can try to analyze the queries and try to give a some readymade

answers, and it was interesting that after some training, the chatbot was so good that the students failed to notice.

(Refer Slide Time: 09:57)

**And Goals can be Practical: Domain-specific Chatbots**

Jill wasn't very good for the first few weeks after she started in January, often giving odd and irrelevant answers. Her responses were posted in a forum that wasn't visible to students.

"Initially her answers weren't good enough because she would get stuck on keywords," said Lalith Polepeddi, one of the graduate students who co-developed the virtual TA. "For example, a student asked about organizing a meet-up to go over video lessons with others, and Jill gave an answer referencing a textbook that could supplement the video lessons — same keywords — but different context. So we learned from mistakes like this one, and gradually made Jill smarter."

After some tinkering by the research team, Jill found her groove and soon was answering questions with 97 percent certainty. When she did, the human TAs would upload her responses to the students. By the end of March, Jill didn't need any assistance: She wrote the class directly if she was 97 percent positive her answer was correct.

1

<sup>1</sup><http://www.news.gatech.edu/2016/05/09/artificial-intelligence-course-creates-ai-teachers>

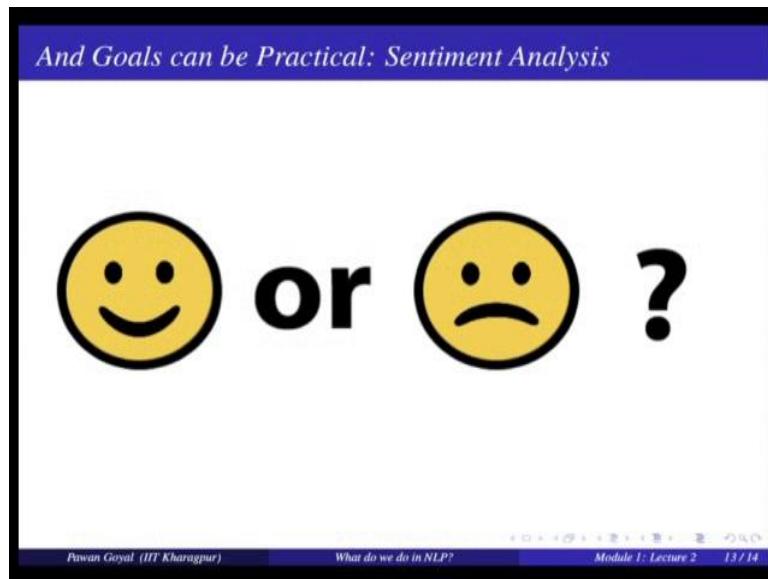
Pawan Goyal (IIT Kharagpur)      What do we do in NLP?      Module 1: Lecture



So, from the article if you see that after sometime when the chatbot learnt from the way issues for querying and the responses that were ideal for the queries, it was giving answers with some roughly like 97 percent certainty and TA is the actual, the human TA were actually check the responses first and then they will upload on the portal.

So, this was again very very practical goal. So, you can contrast it with the open domain chatbot that we talked about. So, this was very very domain specific chatbot. So, they built it only for their own course. So, domain was fixed to their course and the kind of queries you can expect are also limited in number fixed to a certain domain. So, building these domains specific chartbots or conversation agent is it practical very very practical goal and also coming up to be important application in recent days. So, thinking, starting from the conversation agent that can help you by some product on an e-commerce website, instead of you having to search everything; can you just provide your specifications to the chatbot and it can search a product for you or in the case of any flight booking system or banking system where you can give your queries and the chatbot can come up with the possible reply by looking into the document and so on. And this is the practical application that can be solved using NLP.

(Refer Slide Time: 11:28)



Then there is a problem of sentiment analysis again lot of work has gone in NLP on this and this is again a very very practical code. So, from your all your tweets, all your opinions and comments that you provide in social media, can a tool find out what are various sentiments of users and with that you can also find out are they some transitions and sentiments of the users over the years; a lot of research was done with recent presidential elections and in India the Lok Sabha elections, a lot of research was on finding out what are people's opinions and sentiments about various political parties and leaders; many of them actually came across to even predicting who will be the winner of the elections.

(Refer Slide Time: 12:18)

Other Goals

- Spam detection
- Machine Translation services on the Web
- Text Summarization
- ...

Natural Language Technology not yet perfect  
But still good enough for several useful applications

Pawan Goyal (IIT Kharagpur)      What do we do in NLP?      Module 1: Lecture 2      13 / 14

And there are many many other goals. So, we talked about some interesting goals like building say sentiment analysis, building domains, specific conversational agents, doing query completion or auto correction of the query, but there are many other goals like spam deduction. So, you will see that if you are using Gmail or any other web service many of the emails going to the spam folder directly without even bothering you. So, what is happening at the back end? Once an email comes in, so the system tries to see if it is spam or not by doing again text analysis over there and if it is spam it is not even shown to you in your inbox it is directly sent to some spam folder. So, spam deduction is again a very practical goal in not only in your emails also on with social media even on tweets, YouTube comments, even YouTube videos finding out what are its spams is again very interesting and challenging problem.

Then you have the problem on machine translation services on the web. So, think about opening a web page from some other country. So, like from China or Japan suppose you are going to visit that country and you want to read that page. So, you can use Google translate to load that page in English or in any other language for you and that is really helpful. So, again this is a very practical application where NLP is used. And finally text summarization, so given a big news article or scientific article can I summarize that in short; and then there are many many other applications where NLP is actually used.

So, remember one of the some of the previous slides that we saw in this lecture. So, we found that NLP technology is not perfect. So, there are many places where the systems make blunders. So, it is not perfect, but it is still good enough for many many good applications. So, you can know that by, the way you are using it in your daily life. So, you can still use that in many day to day life applications and that is what is guiding this field that so there are lot of applications and that you can think of. So, can you come up with nice ideas, nice algorithms for solving that problem and actually people would use that and benefit from that? So, lot of applications you can think off where you can help the society by doing text processing and analytics.

So, in this lecture we discussed what are some of the things that we do in NLP. So, next lecture onwards we will start talking about see why is NLP hard, what makes the language processing a difficult task to handle.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 03**  
**Why is NLP Hard**

Hello everyone. So, welcome to the 3rd lecture of the first week. So, in the last lecture we were discussing, what do we do in NLP, what are the various applications where NLP has been used and in which been use currently, in what are some of the future potentials of NLP. So, today in this lecture, we will discuss why is NLP hard, what are the some of the difficult is that we face while designing algorithms for NLP and why do we need to worry about various machine learning techniques and all in NLP.

(Refer Slide Time: 00:50)

The slide has a dark blue header bar with the text "Why is NLP hard?" in white. Below the header is a light purple rectangular area containing the following text:

*Lexical Ambiguity*

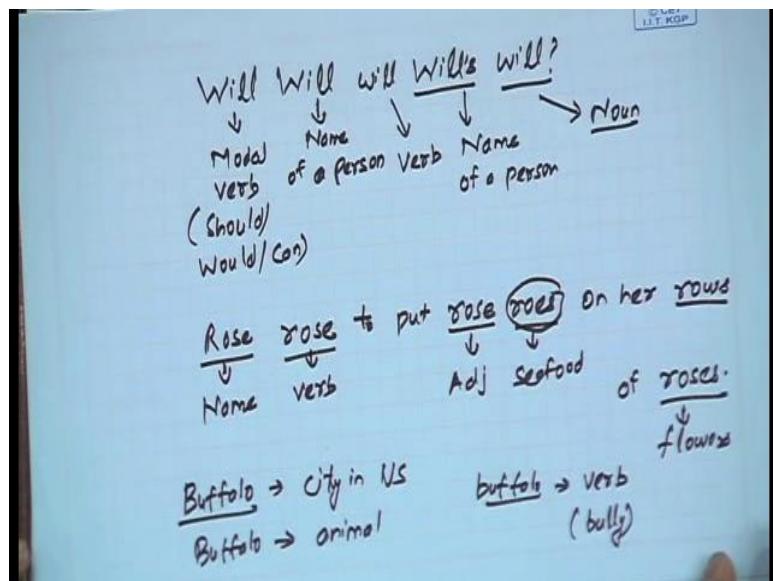
- Will Will will Will's will?
- Rose rose to put rose roes on her rows of roses.
- Buffalo buffalo Buffalo buffalo buffalo Buffalo buffalo.

At the bottom of the slide, there is a dark footer bar with the following text:  
Pawan Goyal (IIT Kharagpur)    Why is NLP hard?    Week 1: Lecture 3    2 / 16

So, which is start with a very simple case of the kind of problems that you face in NLP; we start with the case of lexical ambiguities; ambiguities as you understand is implied for the same word, meaning various interpretations.

So, take this example sentence here, so I have this sentence will will will will's will. So, what you see here is the same word will, has been used 5 times.

(Refer Slide Time: 01:20)



So, let us try to find out what are the various interpretations of meaning the same what will has been used for; will will will will's will. So, what do you say about the first will? So, the first will is a modal verb like should, would, can etc; the second will here is a name of a person; this sometimes easy to get from the English sentence because this is always been in capitals; then you have the third will, this is a verb in the sentence to express his will; the fourth one is again a name of a person you can see it by the apostrophe that is after this word, but it can be either the same will or a different will. So, again is a name of person, but we don't know it is the same will or a different will.

But by the way, the way we write language and the way we communicating language we can say this probably is a second person because if it by the same person I would have used probably hinge and this will is a noun, the will itself. So, what you are saying in the same sentence, will the person will, will express or use will's will, we are using the same word will at least in four different meanings. So, this is one of the extreme cases that we see in language, but let us try to go through some other examples.

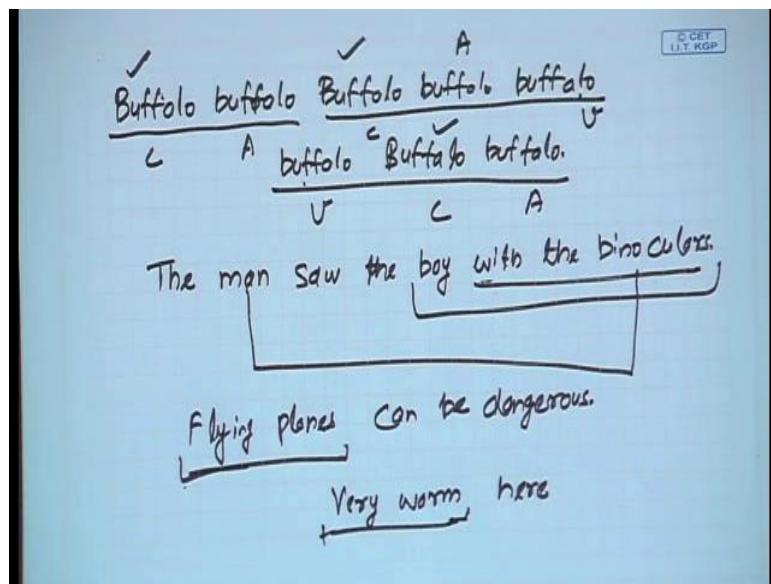
So, this is another example: rose rose to put rose roes on her rows of roses. So, here can you try to decipher what are the various meanings that I are implied (Refer Time: 03:17) for the same word rose. So, what you say about the first rose? The first rose is the name of the person, rose this should be a verb in the past tense to put rose roes, so what is that? This will be an adjective and rose is some sort of the seafood and then we have the next sentence on

her rose of roses and these versus you can find out these are the flowers and this is a rose with you.

So, again we see here the same word rose, if you see the way it is written, in terms of autography the same word rose has been used one now as a name; as a verb, as an adjective and as a flower – 4 different senses. But for the time just try to think of some speech recognition system, we are trying to pronounce this sentence and it has to transcript what are the different words that you have said in your utterance. So, it will have ambiguity, even to find out whether this rose means r o e s or r o s e; this is another problem that this is also handled by various models that we will see in natural language processing.

And let us take another extreme example. So, the Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo. So, now, this sentence is composed only by a single word used 8 times. So, can you try to decide for this? So, let me give you a hint here. So, here word buffalo has been used in 2 sense 3 senses Buffalo, one is the city in US, another Buffalo is an animal and the third buffalo is like a verb that is used in the same sense as bully. So now given these three interpretations, can you try to decipher the meaning of the sentence?

(Refer Slide Time: 06:00)



So, let us try to identify various blocks here; Buffalo buffalo Buffalo buffalo buffalo Buffalo buffalo. So, let me try to give you what are the units together ok.

So, there are 3 units in this sentence. So, now, what would be the interpretation? So, these are the cities by C and the animal A, A and A, and these 2 are verbs. So, then this would be the interpretation of the sentence, buffaloes from buffalo Newyork whom buffaloes from buffalo Newyork bully, bully buffaloes from buffalo Newyork. Probably is not the sentence that you will encounter very often in the corpus, but this is just to convey a point that in language you can actually use the same word and multiple different meanings and that creates a problem and this problem is called as Lexical ambiguity, the same word in lexicon is used in multiple different senses.

(Refer Slide Time: 07:46)

The slide has a dark blue header bar with the title "Why is NLP hard?" in white. Below the header, there are two main sections, each enclosed in a light purple rounded rectangle:

- Language ambiguity: Structural**
  - The man saw the boy with the binoculars.
  - Flying planes can be dangerous.
  - Hole found in the room wall; police are looking into it.
- Language imprecision and vagueness**
  - It is very warm here.
  - Q: Did your mother call your aunt last night?  
A: I'm sure she must have.

At the bottom of the slide, there is a dark footer bar with the text "Pawan Goyal (IIT Kharagpur)" on the left, "Why is NLP hard?" in the center, and "Week 1: Lecture 3 3 / 16" on the right. There are also small navigation icons in the footer bar.

So, now let us go to the next problem or again get to an ambiguity that is a structural ambiguity; what do I mean by structural ambiguity? So, I can have different interpretations of the same sentence. So, let us see the first sentence here. So, this is very very common example that we given NLP, the man saw the boy with the binoculars. So, can you try to see what is the ambiguity here? So, the ambiguity here is whether the binoculars are with the boy or the binoculars are with man; whether the men saw the boy with this binoculars or whether the men saw the boy who was standing with these binoculars. So, these are two different interpretations of the same sentence.

Let us see the other sentence here flying planes can be dangerous; can you see the two interpretations of the sentence, what is dangerous? Are the planes dangerous or flying dangerous? You see flying planes can be dangerous. So, whether the flying is dangerous or

flying planes together is dangerous, these are the two interpretations of the same sentence. Similarly you can see a third sentence here - hole found in the room wall police are looking into it. So now, can you see the ambiguity in the sentence? So, ambiguity here is what are police looking in to or the looking into the hole or the looking in to the matter that there was a whole and we are looking in to the matter.

So, this is another problem that we face very very often with language, that this is ambiguous both in terms of lexical ambiguity; the same word can imply multiple meanings or a structural ambiguity where the same sentence can be interpreted in multiple ways.

And then we have some other problems like languages very imprecise and vague. So, what are the examples here? So, here is simple sentence, it is very warm here. So, can you see what is the vagueness or imprecision here? So, whenever I see a word like it is very warm here, I cannot tell for sure what would be the temperature there? If I am in India, for me warm may mean a temperature of only 40 degrees Celsius, but if I am in UK Europe for me warm might mean 25 degree Celsius. And it might also depend on what was the weather in the last month and so on. So this depends on a lot of context you find out what is the actual temperature that is been conveyed by this simple sentence.

Similarly if you see the other example; so have a question, did your mother called your aunt last night and the answer is I am sure she must have. So, what is the imprecision and vagueness here in the sentence? So, you see whenever I say I am sure she would have done that; that means, I do not know; if I know that I will say yes she has called, but whenever I am saying I am sure she must have. So, probably I do not know whether she has.

(Refer Slide Time: 10:53)

The slide has a blue header bar with the text "But that's the fun part of it". The main content area is white. A question is displayed: "Why is the teacher wearing sun-glasses?". Below it is the answer: "Because the class is so **bright**.  
..."

At the bottom left is the text "Ptwan Goyal (IIT Kharagpur)". In the center is "Why is NLP hard?". At the bottom right is "Week 1: Le". There is also a small circular profile picture of a man in the bottom right corner.

That is what I say this is the fun part of NLP with that helps in constructing a lot of jokes.

So, I have the symbol, this is a nice joke for the classes, so why is the teacher wearing sun glasses and you can given answer because the class is so bright and you can see the bright might mean either the class is bright, in the sense of lot of sun light and on or because the class is very bright in the sense of the students being really intelligent, fine.

(Refer Slide Time: 11:26)

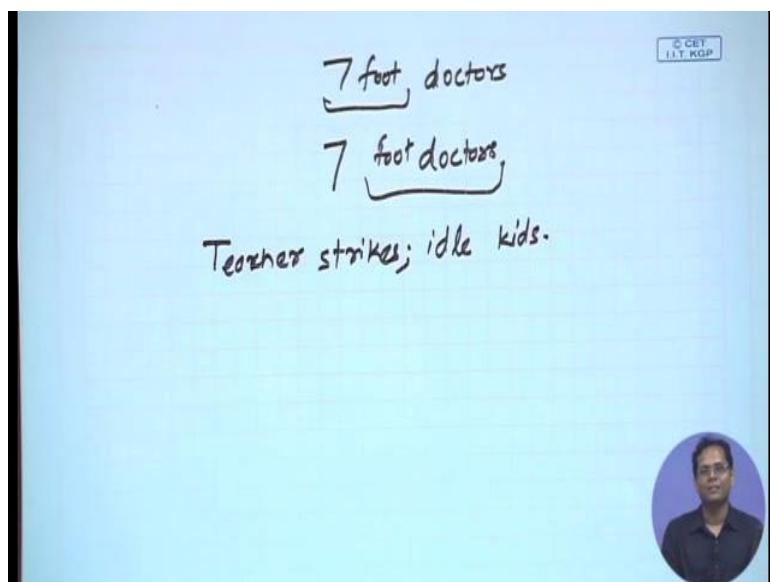
The slide has a blue header bar with the text "Ambiguities". The main content area is white. A section titled "News Headlines" contains a list of three items:

- Hospitals Are Sued by 7 Foot Doctors
- Stolen Painting Found by Tree
- Teacher Strikes Idle Kids

At the bottom left is the text "Ptwan Goyal (IIT Kharagpur)". In the center is "Why is NLP hard?". At the bottom right is "Week 1: Le". There is also a small circular profile picture of a man in the bottom right corner.

So, continuing on the same topic of ambiguities. So, let us see some other examples. So, that is something that we see in news headlines, which is the first sentence: hospitals are sued by 7 foot doctors, so can we see the ambiguity here? In general when you look at this sentence what comes to your mind? Probably the doctors as 7 foot, 7 foot doctors of course, we will 7 feet doctors, but yes this might you want to a interpretation, but what is implied by the sentence? There are 7 different doctors and they are all 4 doctors, so that is what is implied 7 different foot doctors.

(Refer Slide Time: 11:47)



Take the next sentence here, stolen painting found by tree. So, can you see the ambiguity? So, it looks as if the tree found the paintings, but what it means is that the paintings was found near the tree. Take the third example, teacher strikes idle kids. So, when you see the sentence as it is what comes to your mind? Teacher strikes some kids, but what the headline were meaning teacher is striking there is some semicolon and the kids are idle.

(Refer Slide Time: 12:55)

Ambiguity is pervasive

- Find at least 5 meanings of this sentence:
  - ▶ I made her duck
- I cooked duck for her
- I cooked duck belonging to her
- I created the (artificial) duck, she owns
- I caused her to quickly lower her head or body
- I waved my magic wand and turned her into a duck

Pawan Goyal (IIT Kharagpur) Why is NLP hard? Week 1: Lecture 3 6 / 16

So, let us take let us do a simple exercise on the same topic of ambiguity. So, I give you a simple sentence are made her duck. Now try to find, peoples there are 10 or more than 10 meanings for the sentence, but try to find at least 5 meanings of this sentence, I made her duck. So, let us do this exercise.

(Refer Slide Time: 13:20)

I made her duck.

↓  
Cook/make

→ ditransitive  
transitive

1. I cooked a duck for her. / dative

2. I cooked a duck belonging to her. possessive

3. I made the (artificial) duck she owns.

4. I made her lower her head. Action trans.

5. I waved my magic wand that turned her into a duck.

© CSET  
I.I.T. KGP



I made her duck. So, what are different meanings this sentence can take? So, we need to see what are the different interpretations each of the word can have in this sentence; so for example, the word made can mean for example, cook or make. So, one interpretation can be I

cooked a duck for her. So, simple interpretation I cooked a duck for her that is one interpretation.

Now, in the same sense of cooking, I can also try to write a different interpretation what is that? So, one meaning is I made her duck another could be I made a duck that belong to her. I cooked a duck belonging to her. So, that is her duck, it may not, I might not have cooked for her, I might have cooked for myself, but I cooked the duck that belong to her I made her duck. Now made can also mean is simple making, so you can think of an artificial duck like a toy and I can say I made the artificial duck she owns of course, you can also have the second interpretation here, I made the artificial duck for her or belonging to her.

But let us take this interpretation, now what are the other two interpretations you can think of? So, now, try to think of the other meaning of duck, can duck be used as a verb? So, if you are listening to some cricket commentaries sometimes, the batsman duck whenever they were bouncer in sometimes so; that means, (Refer Time: 12:25) once head. So, one interpretation can be I made her lower her head that can be another interpretation. Now can you think of any different interpretation from all the four that we have seen till now? So, the hint is that try to go in the Harry potter mode. So, this is something like I waved my magic wand and converted her into duck, yes that is a possible interpretation, I waved my magic wand that turned her into a duck ok.

So, you see the simple sentence I made here duck can have at least these 5 different interpretations; so these are the 5 interpretations that we saw. So, now, what is in the language that gives rise to all these different interpretations? So, let us try to look closely.

(Refer Slide Time: 16:37)

Ambiguity is pervasive

**Syntactic Category**

- 'Duck' can be a noun or verb
- 'her' can be a possessive ('of her') or dative ('for her') pronoun

**Word Meaning**

- 'make' can mean 'create' or 'cook'

Pawan Goyal (IIT Kharagpur)      Why is NLP hard?      Week 1: Language Processing



So, this ambiguity is pervasive everywhere, so how? So, one thing is about the syntactic category; what is the role of a word in a sentence. So, you see the word duck here it can mean either a noun or a verb. So, can you label the sentences here, where duck has been used as a noun? So, this is noun, this is noun, this is noun, this is verb, this is noun. So, fine, these are two interpretations that can be noun or a verb.

Then there is a case with her, the word her can either be a possessive of her or dative for her. So, can you see the two examples here: the two interpretations which were made because of these; this was dative I did it for her or this is possessive belonging to her. So, these are again two interpretations for this ambiguity in language then we see make can mean either to create or cook this. So, this for cooking and this was for making, what is?

(Refer Slide Time: 17:58)

Ambiguity is pervasive

*Grammar*

make can be

- **Transitive:** (verb with a noun direct object)
- **Ditransitive:** (verb has 2 noun objects)
- **Action-transitive:** (verb has a direct object + verb)

*Phonetics*

- I'm eight or duck
- I'm aid her duck

Ptwan Goyal (IIT Kharagpur) Why is NLP hard? Week 1: Language Ambiguity

Then if you go to grammar, the same work make can be either transitive; that means, it will be a verb with a single noun as direct objective, it can be ditransitive; that means, a verb that is having two different non objects or action transitive, it has a direct object plus a verb.

So, in these 5 interpretations can you try to mark them, where the verb make was uses transitive, ditransitive and action transitive? So here I have the same word make which is having an object and a verb. I made her lower her head or I made her do something. So, this is action transitive; what is ditransitive? So where there are two objects of the same verb; I cooked a duck for her. So, this as two different objects and what is the single transitive? I could they duck belonging to her this is a single object. So, here there are two different objects: this is ditransitive and this is transitive. So, in language the same verb may can be use any of these 3 different vague and that gives me 3 different interpretations.

So, now, suppose you go to phonetics. So, I am speaking this sentence, I made her duck what are different interpretations you can think of? So, what happens in speech recognition? I am a spin something and you have to transcript. So, whenever I see I made her duck you might thing of all these possible transcriptions. So, like I am eight or duck, I am aid her duck all these are possible, but the problem that NLP decisions will face is going there are many different possible interpretations, which going to choose in a given context.

(Refer Slide Time: 20:04)

Ambiguity is Explosive

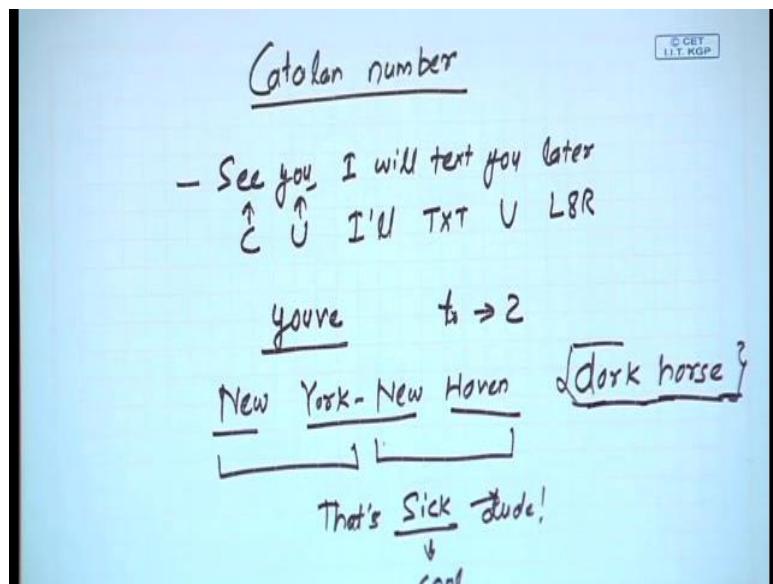
- I saw the man with the telescope. **2 parses**
- I saw the man on the hill with the telescope. **5 parses**
- I saw the man on the hill in Texas with the telescope. **14 parses**
- I saw the man on the hill in Texas with the telescope at noon. **42 parses**
- I saw the man on the hill in Texas with the telescope at noon on Monday. **132 parses**

Pawan Goyal (IIT Kharagpur) Why is NLP hard? Week 1: Language Processing

So, let see something from the (Refer Time: 20:07) ambiguity now. So, I have the simple sentence I saw the man with the telescope and we saw earlier it can have two different parses. So, parses I mean different ways in which the words can be connected in this sentence.

So, we will have a complete topic on parses is give we will discuss there in detail, but I guess you can see at least the idea from the last example. So, now, suppose I try to increase the length of the sentence, I saw the man on the hill with the telescope and immediately you can find 5 different parses of the sentence; this does not stop here. So, let me have this sentence I saw the man on the hill in Texas with the telescope. So, now, it has 14 parses, if I say I saw the man on the hill in Texas with the telescope at noon it has 42 parses and if I say I saw the man on the hill in Texas with the telescope at noon on Monday it look 132 parses and you can actually relate these numbers to something call a Catalan number.

(Refer Slide Time: 21:06)



So, as you keep on increasing the number of phases in the sentence, the number of interpretations in which they can be connected in the sentence increase.

(Refer Slide Time: 21:21)

### Why is Language Ambiguous?

- The goal in the production and comprehension of natural language is efficient communication.
- Allowing resolvable ambiguity
  - ▶ permits shorter linguistic expressions
  - ▶ avoids language being overly complex
- Language relies on people's ability to use their knowledge and inference abilities to properly resolve ambiguities

Pawan Goyal (IIT Kharagpur) Why is NLP hard? Week 1: Lecture 3 10 / 16

So, now why is language ambiguous? This can be a nice question that why is language ambiguous at all. So, we need to understand what the goal of language as such. So, language is used for communication. So, the goal introduction or communication languages to be able to communicate ideas clearly, but at the same time I should certain restrictions or that then that a post like I should have shorter linguistic expressions. So, think of the previous example

that we saw in the last slide, the same sentence was having 132 different interpretations. Now, what if I need to have a different sentence for each of these 132 different interpretations that will make by language expression really large and also the language will become very complex.

So, what happens in language? Some sort of ambiguities allowed, but is ambiguity is some that is easily resolvable; so you cannot have an ambiguity there is not resolvable at all; by some sort of knowledge that you have human being tries to resolve it, but in the case of NLP we try to we have, we have this starts for developing algorithm that kind resolve this ambiguity. So, yes language relies mostly on the people's ability to use their knowledge and some inference capabilities to properly to resolve these ambiguities.

(Refer Slide Time: 22:58)

*Natural Languages vs. Computer Languages*

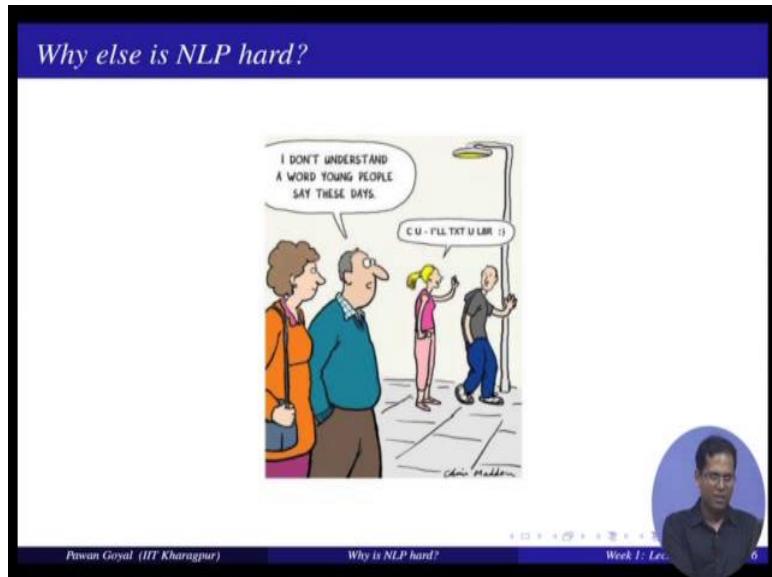
- Ambiguity is the primary difference between natural and computer languages.
- Formal programming languages are designed to be unambiguous
  - ▶ *Formal programming languages can be defined by a grammar that produces a unique parse for each sentence in the language.*
- Programming languages are also designed for efficient (deterministic) parsing.

Pawan Goyal (IIT Kharagpur)      Why is NLP hard?      Week 1: Lec 1

So, now, just a brief discussion on what is the difference between a natural language and a computer language as such; one primary difference is the ambiguity. Programming language do not have any ambiguity in, so whenever you write a program it will mean only one particular thing these no ambiguity there, but that is not the same that is not true for the case of language in natural language.

So, all the programming languages are formal and they are designed to be unambiguous so that you can have very very efficient passing for that. So, they can always defined by a grammar, that process is unique parse for each sentence or each programming construct in the language that is not true for the natural languages, so you can have parsing in linear time.

(Refer Slide Time: 23:50)



Now why else is NLP hard? So, this cartoon gives you some idea that I do not understand a word young people say these days. So, I am talking about social media here. So, see you, I will text you later. The sentence see you, I will text you later now your writing is C U I'll TXT U L8R. So, now the problem at hand is to understand that that you have actually meaning the sentence and find out a starting from C U mean this see, from U the mean this you and so on.

(Refer Slide Time: 24:46)

**Non-standard English**  
Great job @justinbieber! Were SOO PROUD of what youve accomplished! U taught us 2 #neversaynever & you yourself should never give up either

**Segmentation Issues**  
the New York-New Haven Railroad  
*the* [New] [York-New] [Haven] [Railroad]  
*the* [New York]-[New Haven] [Railroad]

The slide has a blue header and footer. The footer includes the text "Pawan Goyal (IIT Kharagpur)", "Why is NLP hard?", "Week 1: Lec.", and a navigation bar.

So, this is also called the non standard use of English and that is very very prevalent with the use of social media, all the SMS and other platforms.

So, let us see this particular tweet. So, great job Justin Bieber we are so proud of what you have accomplished, you taught us to neversaynever and you yourself should never give up either. So, in this sentence see it is a tweet. So, what are the things that you do not see in formal language? So, one example here is mentioning the use of mention at justinbieber. So, this mention is again wherever is specific to tweet and you are using hash tag neversaynever – this is in the hash tag; what else? So, we are seeing constructs like this, you have is written as you have you v e and to is written as 2. So, this is again non standard use of English that makes NLP difficult. Then there are many other problems like segmentation issue. So, I have the simple sentence the New York-New Heaven Railroad.

So, where should I segment this particular sentence, especially New York-New Haven, should I segmented like that? New York- New Haven or New York, New Haven the second is the correct segmentation. So, there are two possible segmentation of the same sentence.

(Refer Slide Time: 26:31)

The slide has a dark blue header with the title 'Why else is NLP hard?'. Below the header, there are two main sections: 'Idioms' and 'neologisms'. The 'Idioms' section contains three items: 'dark horse', 'Ball in your court', and 'Burn the midnight oil'. The 'neologisms' section contains three items: 'unfriend', 'retweet', and 'Google/Skype/photoshop'. At the bottom of the slide, there is footer text: 'Ptwan Goyal (IIT Kharagpur)', 'Why is NLP hard?', 'Week 1: Lecture 3', and '14 / 16'. There are also small navigation icons for a presentation slide.

So, problem with NLP would be to find out what is the correct segmentation given this sentence, then there are other cases like the case of idioms in language. So, what happens in the case of idioms you cannot construct the meaning of the page by looking at the meaning of the individual words and trying compose them together. So, if I say idiom like dark horse, I cannot take the meaning of a horse which is dark and make the meaning of this idiom; dark

horse is something else, so this is some sort of idiom that is conveyed for the person who is not well known, but he suddenly becomes dark horse. So, the he is suddenly excelling in certain field.

Similarly you see this idiom ball in your court, does not mean the ball is in your court; that means the matter is at your hand and now it is your turn. Similarly this idiom is again very much used burn the midnight oil, there is not mean that I am burning the oil at midnight, but what it means is that you are doing hard work. Then there are various new words that have been floated new usages that are coming up, so they also called new (Refer Time: 27:49). So, for a one particular example, these are examples are taken from social media is unfriending. I have some facebook friend and I am unfriending. So, this has become is a verb itself; retweet - retweet has been used a lot as a verb, similar Google, Skype, Photoshop all are very much used in as verbs and Googling and all.

So, that creates another problem for NLP that is new words are coming up in different and new usages, so do not have a closed vocabulary and your vocabulary keeps on increasing.

(Refer Slide Time: 28:29)

The slide has a blue header bar with the text "Why is NLP hard?". Below the header is a white main content area. On the left side of the content area, there is a light purple callout box containing the text "New Senses of a word" in blue. Below this text is a bulleted list:

- That's sick dude!
- Giants

At the bottom of the slide, there is a dark blue footer bar with the following text:

Pawan Goyal (IIT Kharagpur)      Why is NLP hard?      Week 1: Lecture 3      15 / 16

Then you keep on getting new senses of the words. So, see this particular example - That's sick dude! So, I am pointing here to the word sick. So, what is the usual meaning of sick that you see? Sick I will something that is not healthy, that is someone who is ill, but in this particular sentence that is sick dude, sick mean sick means something that is cool and that meaning is coming up with social media. Similarly you see the word giants, what is the

particular meaning of giants seasonally we see giants as some sort of demons that we have in storage. But recently some new meaning of giants is coming up, so that you can think of, like some sort of giant multi nationals and all that.

(Refer Slide Time: 29:20)

The slide has a blue header bar with the text "Why is NLP hard?". Below it is a white main area containing two purple callout boxes. The first box is titled "New Senses of a word" and contains the following bullet points:

- That's sick dude!
- Giants ... *multinationals, conglomerates, manufacturers*

The second box is titled "Tricky Entity Names" and contains the following bullet points:

- Where is *A Bug's Life* playing ...
- *Let It Be* was recorded ...

At the bottom left is the name "Piwan Goyal (IIT Kharagpur)". In the bottom center is the slide title "Why is NLP hard?". At the bottom right is a circular profile picture of a man and the text "Week 1: Lec. 6".

Giant manufactures and then there is a problem of entity names. So, take this sentence where is a bug's life playing? You cannot understand the meaning of the sentence unless you know that this particular utterance a bugs life is single entity and then you try to understand the meaning of the sentence. Similarly the second sentence let it be was recorded you need to understand that let it be is single entity here.

(Refer Slide Time: 29:52)

The slide has a blue header bar with the title "What we do in NLP?". Below the header, there are two main sections:

- Tools Required**
  - Knowledge about language
  - Knowledge about the world
  - A way to combine knowledge resources
- How is it generally done?**
  - Probabilistic models built from language data
    - $P(\text{"maison"} \rightarrow \text{"house"})$  is **high**
    - $P(\text{I saw a van}) > P(\text{eyes awe of an})$
  - Extracting rough text features does half the job.

At the bottom of the slide, there is footer text: "Piyan Goyal (IIT Kharagpur)", "Why is NLP hard?", "Week 1: Lecture 3", and "16 / 16". There are also small navigation icons at the bottom right.

So, what we do in NLP to handle all that? So, we need to have some knowledge the language how the sentence is constructed, what kind of words are there and so on.

We need to knowledge about the world and we need to have the way in which we will combine various knowledge resources in an efficient manner. So, how is it generally done? So, most of the times we do it by using various probabilistic models; this is the single simple example here. So, I have word like [FL] in French and may translate in English and there are multiple interpretations, so I want to use a model that (Refer Time: 30:30), the probability that [FL] goes to house is high and I should choose this particular interpretation. Similarly for the next sentences suppose this is for this speech recognition system, whenever I am saying I saw a van and there are two interpretations I saw a van or eyes of an I should be able to say that the first interpretation is more probable to occur in the language then the second interpretation.

And we will deal with this a lot. So, many a times we have to extract a lot of text a feature that does most of this job, fine. In this lecture we covered some aspects of wild languages hard. So, in the next lecture we will start talking about some of the very very basic empirical laws that we see in language and we will start with doing some basic pre processing.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 04**  
**Empirical Laws**

So, welcome back for the fourth lecture of the first week. So, in the last lecture we discussed why is NLP hard and we took examples of various cases of ambiguities in the language and some non standard usage also. So, (Refer Time: 00:36) we will actually go into a real text corpus and try to study some very simple empirical laws, that are very very universal about language.

(Refer Slide Time: 00:52)

**Function Words vs. Content Words**

Function words have little lexical meaning but serve as important elements to the structure of sentences.

*Example*

- The *wifly prunkilmonger* from the *glidgement mominkled* and *brangified* all his *levensers vederously*.
- Glop angry investigator larm blonk government harassed gerfritz infuriated sutbor pumrog listeners thoroughly.*

*Function words are closed-class words*  
prepositions, pronouns, auxiliary verbs, conjunctions, grammatical articles, particles etc.

Pawan Goyal (IIT Kharagpur)      Empirical Laws      Week 1: LA

So, now before going to that, I will try to make a simple distinction clear to you. So, what are the difference between function words versus content words in language. So, whenever you see a language in its vocabulary, there are various kinds of words and they can be called either function words or content words what are the difference between those?

So, function words are mainly used to make the sentence grammatical. So, things like where is determine is, a and pronouns and all are function words; content words are various nouns and verbs that convey what are the important concepts in the sentence. So, content words are mainly used for determining the structure of the sentence. So, now, this is an example to give you an idea about what are content words and what are function words.

So, you see here the same sentence has written in 2 different ways, in one of the cases we have replaced the content words by some garbage words. So, this mean (Refer Time: 02:03) the words that are not in the language.

In other example we have (Refer Time: 02:06) the function words by some arbitrary words that are not in the language. So, can you try to find out in which sentence we have replaced the function words with something else? So, we will see here in the second sentence, we have removed the word like the and replaced with words like glop and so on. In the first sentence we have replaced all the nouns like angry and investigator, with winfy prunkilmonger that is research to the garbage word that is not the language, now.

Now, try to see the 2 sentences end, in which sentence you can see the structure of the sentence clearly and in which of the sentence you can understand the topics of the sentence clear and you will find out that in the second sentence you know all the topics. So, this is talking about some investigator, government and infuriated and all that, the information you cannot get from the first sentence. From the first sentence you can understand the structure of the sentence, the some x some noun from something and a verb and again a verb all his a noun and a adverb. And you do not know what are these nouns and adverbs, but you know what is the structure of these sentence, there are certain fillers in the sense of various nouns and verbs.

So, in general function words in a language are closed class words, what I mean by that there are very specific grammatically categories that can be called as function words and they are generally closed, we do not have newer and newer functions what coming into a language. Some examples of function words are like prepositions, pronouns, auxiliary verbs, conjunctions, grammatical articles and various particles in the language.

(Refer Slide Time: 04:04)

*Most Common Words in Tom Sawyer*

Word	Freq.	Use
the	3332	determiner (article)
and	2972	conjunction
a	1775	determiner
to	1725	preposition, verbal infinitive marker
of	1440	preposition
was	1161	auxiliary verb
it	1027	(personal/expletive) pronoun
in	906	preposition
that	877	complementizer, demonstrative
he	877	(personal) pronoun
I	783	(personal) pronoun
his	772	(possessive) pronoun
you	686	(personal) pronoun
Tom	679	proper noun
with	642	preposition

The list is dominated by the little words of English, having important grammatical roles.

Pawan Goyal (IIT Kharagpur)      Empirical Laws      Week 1: Language



So, now let us take a corpus, and try to see what are the distribution of words, that we encounter in the corpus. So, here we have taken Tom Sawyer. So, that was a noble written by Mark Twain and what we are doing in the slide, we are just reporting which of the words occur with what frequency; so this is a sorted list starting from the highest frequency to some top 15-20 words and what is the grammatical category. So, the top word you see here is ‘the’, so that occurs with the frequency of 3332 and this is very very common across different language corpus. So, if you pick up any arbitrary language corpus, you will feel that the distribution is roughly the same

So, now can you have a look at this list and find out, what is special about this list, what is the category of words that you see here? So, remember we talked about 2 categories: function words and content words, can you try to find out which of the category you see here? So, we see most of the words here are function words. So, the, and, a, to, of, and you can see also by the grammatical categories that is also written here. So, one thing like that we see in this distribution is that this list is dominated by the little words of English that are having very important grammatical roles for the sentence.

Now, what is one exception that you see to this? So, we call them function words that we said earlier determiners, prepositions, complementizers like the word that here.

(Refer Slide Time: 05:54)

*Most Common Words in Tom Sawyer*

Word	Freq.	Use
the	3332	determiner (article)
and	2972	conjunction
a	1775	determiner
to	1725	preposition, verbal infinitive marker
of	1440	preposition
was	1161	auxiliary verb
it	1027	(personal/expletive) pronoun
in	906	preposition
that	877	complementizer, demonstrative
he	877	(personal) pronoun
I	783	(personal) pronoun
his	772	(possessive) pronoun
you	686	(personal) pronoun
Tom	679	proper noun
with	642	preposition

The one really exceptional word is *Tom*, whose frequency reflects the text chosen.

Pawan Goyal (IIT Kharagpur) Empirical Laws Week 1: Lecture 4 3 / 15

So, now what is one exception that we see in this list? So, that exception is probably the word Tom; so Tom is very specific to the topic of this text. So, this is about Tom Sawyer. So, you see the word Tom also has a very high frequency and this makes you to the list of top frequent words. So, again if you take an arbitrary text and try to find out some top frequent words, mostly they will be function words; also sometimes known as various stop words, plus you can find some occurrences of words that convey the topic of that text. So, in this that is a word Tom.

(Refer Slide Time: 06:34)

*Most Common Words in Tom Sawyer*

Word	Freq.	Use
the	3332	determiner (article)
and	2972	conjunction
a	1775	determiner
to	1725	preposition, verbal infinitive marker
of	1440	preposition
was	1161	auxiliary verb
it	1027	(personal/expletive) pronoun
in	906	preposition
that	877	complementizer, demonstrative
he	877	(personal) pronoun
I	783	(personal) pronoun
his	772	(possessive) pronoun
you	686	(personal) pronoun
Tom	679	proper noun
with	642	preposition

How many words are there in this text?

Pawan Goyal (IIT Kharagpur) Empirical Laws Week 1: Lecture 4 3 / 15

So, let us try to see something else, how many words are there in this text, what are the various technical terms by which these are indicated?

(Refer Slide Time: 06:48)

## Type vs. Tokens

**Type-Token distinction**

Type-token distinction is a distinction that separates a concept from the objects which are particular instances of the concept

**Type/Token Ratio**

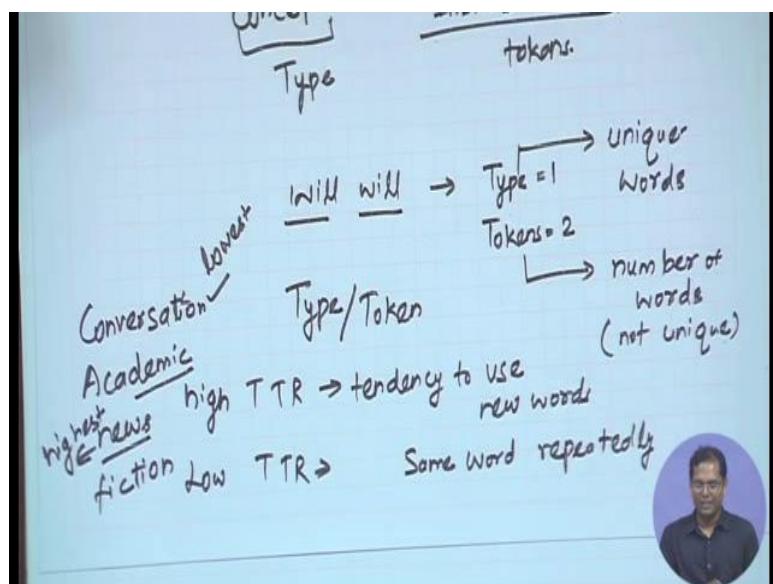
- The type/token ratio (TTR) is the ratio of the number of different words (types) to the number of running words (tokens) in a given text or corpus.
- This index indicates how often, on average, a new 'word form' appears in the text or corpus.



Piwan Goyal (IIT Kharagpur) Empirical Laws Week 1: Le

So, for that we will talk about the type token distinction. So, type token astrologically philosophical distinction between a concepts; concept is a type and a particular instances of the concept. So, concept is type and instance is concepts or tokens.

(Refer Slide Time: 07:05)



Handwritten notes on Type/Token Ratio:

- Type:  $\frac{\text{unique words}}{\text{total words}}$
- Tokens:  $\frac{\text{number of words}}{\text{(not unique)}}$
- Type/Token Ratio =  $\frac{\text{Type}}{\text{Tokens}}$
- Conversation: lower TTR
- Academic: high TTR → tendency to use new words
- News: highest TTR → Some word repeatedly
- Fiction: Low TTR → Some word repeatedly



So, what do I mean by type and token, when it comes to a language? So, language if the same word occurs multiple times, so I call them the same type, same concept, but different tokens. So, if I write the same word say, will will the same word twice, I say this is a single type this is only one type, but there are 2 tokens. So, each individual occurrence is a different token, but they have the same type. So, in other sense by type I mean some sort of unique words in my vocabulary and token means the number of words they are not unique. So, if the word occurs 5 times I will count it as 5 different tokens, but the same type.

Now, so once we have understood what is a distinction between type and token, there is a particular concept that is called type token ratio that is used to should describe some text. So, what is type token ratio? Very simply, the ratio of the number of types that is number of different unique words to the number of running words so all the tokens. So, you find out the number of types in the text, divided by the number of tokens in the text and you get the TTR, Type Token Ratio.

So, what this ratio indicates? That tells on an average how often a new word appears in the text. So, if type divide by token that is my type token ratio if it is high; that means, in the text lot of new words are keep on coming, because the number of types is very close to the number of token, but if this is small; that means, the small words are getting repeated again and again in the corpus.

(Refer Slide Time: 09:37)

The slide has a blue header bar with the text "Comparison Across Texts". Below the header, there are two sections, each with a purple background and white text. The first section is titled "Mark Twain's Tom Sawyer" and contains the following data:

- 71,370 word tokens
- 8,018 word types
- TTR = 0.112

The second section is titled "Complete Shakespeare work" and contains the following data:

- 884,647 word tokens
- 29,066 word types
- TTR = 0.032

At the bottom of the slide, there is a dark footer bar with the following text:

Pawan Goyal (IIT Kharagpur)      Empirical Laws      Week 1: Lecture 4      5 / 15

So, suppose we take 2 different text corpus: one is Mark Twain's Tom Sawyer, another we take complete Shakespeare work and we try to find out what is the TTR for each of these. So, our Mark Twain's we find that there are 71,370 word tokens and 8,018 word types. So, how do we complete TTR? You just find out, you just divide types by tokens. So, 8018 divide by 71370 that will give me the TTR that is close to 0.112.

Now, if you take the Shakespeare work, you find similarly there are 88400 plus tokens and 29000 plus types and type TTR comes to be much more than that of Tom Sawyer, 0.03.

(Refer Slide Time: 10:30)

*Empirical Observations on Various Texts*

*Comparing Conversation, academic prose, news, fiction*

- TTR scores the lowest value (tendency to use the same words) in conversation.
- TTR scores the highest value (tendency to use different words) in news.
- Academic prose writing has the second lowest TTR.

*Not a valid measure of 'text complexity' by itself*

- The value varies with the size of the text.
- For a valid measure, a running average is computed on consecutive 1000-word chunks of the text.

Pawan Goyal (IIT Kharagpur)      Empirical Laws      Week 1: Lec 1

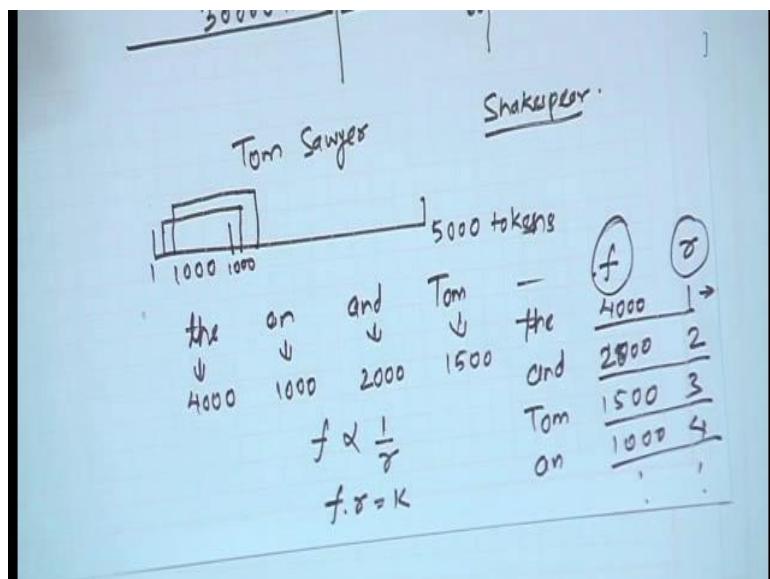
Suppose I take 4 different types of media. So, 4 different types of media like conversations if 2 people are conversing; academic prose will be scientific articles and books; news and fiction; I take 4 different sorts of media in which language is communicated. Now, can you try to have a guess which one will have the highest TTR, and which will have the lowest TTR?

So, let us go to the definition of TTR. So, high TTR, it will be tendency to use the new words, what about low TTR? If you have low TTR, that is tendency to use same word repeatedly. So, now given that I have 4 different media: one are conversation, then I have academic prose, then I have news and fiction. Now which one do you think will have the tendency to repeat the same word again and again? So, we think over it, in conversation we try to repeat the same word again and again. So, whatever you are conversing, I will try to take it from there and repeat it.

So, conversation have the tendency to use the same word repeatedly so that means, they will have the lowest TTR, on the other hand it has been found in news you have tendency to use newer words, so news this is the highest. So, can you guess which of the remaining one will have the second lowest TTR, where the same words have been used repeatedly and that is actually the academic prose; because academic prose is very very formal, so they are very very particular kind of words and verbs that we use. So, that is why TTR is again found to be low in academic prose.

So, TTR is scores the lowest value in the case of conversations and highest value in news and the academic prose as the second lowest TTR and this also make sense. Now one thing you must be conscious about that TTR in itself is not a very valid measure of finding text complexity, you might say that Ok, if TTR is high text might be complex, but this is not a valid measure in itself. So, why is that so? So think of this scenario, if you are using more and more text what would happen to your TTR ratio?

(Refer Slide Time: 13:45)



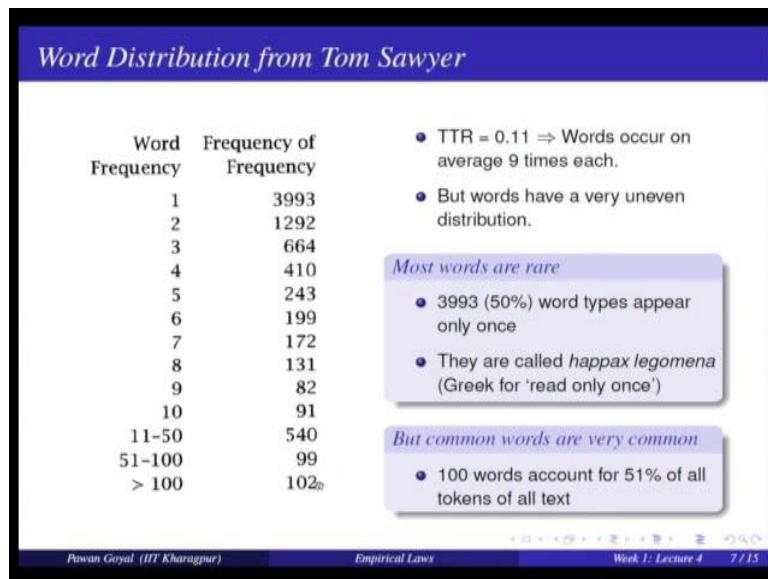
Suppose your text contains, you are taking a running text, so till here you have 30000 tokens, now you go forward in your text and you go till 60000 tokens, now what would you guess, can you see the TTR at this point will always be lower than the TTR at this point? Yes because whatever words were there in the corpus, most the words have already been introduced in the first half; second half most of the words will be again getting repeated. So,

the number of tokens will increase linearly with the size of text, but numbers of words in a vocabulary do not.

So, that is why if I take say a Tom Sawyer versus Shakespeare's works, you will see the TTR is high in Tom Sawyer than Shakespeare work, because they since having a much larger number of tokens. So, in itself it cannot be a valid measure. So, when you take into account the size of the text. So, another way in which this is computed is by taking a running average. So, you take a running average on some consecutive 1000 word chunks, what I mean by running average? Suppose I have a text that has 5000 tokens.

So, now you first compute TTR for the initial 1000 tokens 1 to 1000, get a TTR for that; now you take start from 2 to 1001, I did not have TTR, 3 to 1002 compute TTR and then take an average. So, you compute TTR on moving window of 1000, length 1000 and then you take an average. So, that is considered to be a better measure for TTR.

(Refer Slide Time: 15:51)



So, now, you see that we will talk about some empirical laws, for that let us try to have a look at some of the frequency distribution that we see in Tom Sawyer. So, remember we said that the TTR on Tom Sawyer is roughly 0.11. So, what does it indicate? It says that on an average here what is repeated 9 times, yes try to token ratio is 1 by 9 roughly.

Now, what is that mean, does that mean that each word occurs 9 times in the text or is it something different? So, what we are saying here that words do not have a very even

distribution; it is not happen that each word occurs 9 times in the text. So, what is the usual kind of distribution that we see in the corpus?

So, in the left hand side of the slide, you see I have 2 columns: first column gives me the word frequency, second column gives me frequency of frequency; what do I mean by frequency of frequency? How many words in this corpus have that frequency, so take the first row; that means, there are 3993 words in the corpus, that occur only once they have frequency 1 and there are 1292 words in the corpus that occur twice and so on.

Similarly, there are 102 words in the corpus that occur more than 100 times. So, now, once you see this statistic then immediately see that the distribution is not very very uniform. So, what are the 2 different observations that we have here? So, first observation that we make are most words are real. So, which are the real words? The words that occur with frequency 1 or 2, so we see here roughly 50 percent of the word types; remember in Tom Sawyer roughly 8000 word types were there. So, roughly 50 percent of them occur only once. So, they are the frequency of 1 and they are also called *hapax legomena*. So, that is a Greek term for read only once, they occur only once in my corpus.

What is the other observation that we have? 100 words account for 51 percent of all the tokens of all text. So, you had only 102 words here that were having frequency greater than 100, but the together account for 50 percent of the tokens of the corpus. So, this is again a strange.

(Refer Slide Time: 18:27)

**Zipf's Law**

- Count the frequency of each word type in a large corpus
- List the word types in decreasing order of their frequency

**Zipf's Law**

A relationship between the frequency of a word ( $f$ ) and its position in the list (its rank  $r$ ).

$$f \propto \frac{1}{r}$$

or, there is a constant  $k$  such that

$$f.r = k$$

i.e. the 50th most common word should occur with 3 times the frequency of the 150th most common word.

Pawan Goyal (IIT Kharagpur)      Empirical Laws      Week 1: Lecture 4      8 / 15

So, now this particular observation gives the first empirical law that we will study that is called Zipf's law. So, very very popular law in the case of language corpus, how the vocabulary's distribution behave; so you see Zipf's law what do we need to do? First take a large corpus, now take find out what are the individual word types and count the frequency of each word type.

Now what you need to do? You list the word types in their decreasing order of frequency. So, for example, suppose in the previous case we found there are various types like the, an, and, Tom and so on in the corpus, I will find out the frequency. So, this occurs with 4000 frequency, this occurs with 1000, this occurs with say 2000 and this occurs with 1500 and so on there are various first occurring the different frequencies.

Now, the first thing I will do, I will try to sort them in the decreasing order. So, first I will have - the, then I will have - and, then I have - Tom, then I will have - an, and so on all that the words in the vocabulary. So, they will have their frequency this is 4000, 1000 sorry 2000, 1500 and 1000. Now I will give them the ranks, because I have already sorted them in decreasing order I put the rank. So, there is this rank 1, rank 2, rank 3, rank 4 and so on.

So, now what is Zipf's law say? So, Zipf's law gives me a relationship between the frequency of a word, the  $f$  in this list and the rank of this word, in this sorted list and what is the relation that Zipf law gives? A relation is very very easy. So, relation is the  $f \propto r^{-1}$ , inversely ratio between the frequency and the rank. So, that is I can say that if I multiply  $f \cdot r$ , I will get some sort of the constant  $f \cdot r = k$ ; this is for our arbitrary example that we took in this case we can even see that this holds. So, 4000 into 1 is. So, this is 4000 this is again 4000, this is 4500 and this 4000 they are roughly the same. So, what it means suppose I make a list like that and I have 2 words.

(Refer Slide Time: 21:32)

The image shows handwritten mathematical notes on a light blue background. At the top left, there is a small diagram with two nodes labeled '50' and '150'. Above the first node is the letter 't<sub>1</sub>', and above the second node is the letter 't<sub>2</sub>'. To the right of these, the equation  $f_1 = 3f_2$  is written with a horizontal line underneath it. Below this, the word 'Pr' is written above the fraction  $\frac{f_r}{N}$ . To the right of this fraction, the text 'Total number of tokens' is written with an arrow pointing to the denominator 'N'. Below the fraction, the equation  $f \propto \frac{1}{r}$  is shown, followed by  $f = \frac{A}{r}$  and  $k = A/N$ . There is also a small number '4' at the top right.

So, I have a word with has a rank of 50 and another word that has rank of 150 and I find it's frequency f<sub>1</sub> it is frequency f<sub>2</sub>. So, I can see that f<sub>1</sub> is 3 times f<sub>2</sub>. So, f<sub>1</sub> is 3 times f<sub>2</sub> ( $f_1=3.f_2$ ). So, the 50th most count word should occur with 3 times the frequency of the 150th most convert, this is what Zipf's law tells me. Now, we can write it in also a different manner. So, we talked about frequency, we can also talk about probability. So, let Pr denote the probability of a word with rank r.

So, what is Pr? Pr is nothing but frequency of the word of rank r, divide by the number of tokens that I seen the corpus ( $P_r = F_r/N$ ), that is how I will compute the probability of that. So, N is the total number of word account says tokens. So, what does Zipf law tell? So, it says frequency is inversely proportional to r ( $f \propto r$ ), I can also write f divide by N with some A by r ( $f/N = A/r$ ). So, what I have done? I taken the constant k, I have written that it as A times N ( $k = A.N$ ). N is a number of tokens in my vocabulary N A is from other constant and it is fixed given a corpus and using in the constant. So now, why we are doing that? Because it has been seen in that in the corpus, A's when we take a valid close to 0.1, this is generally more fix than the value of k.

(Refer Slide Time: 23:40)

Empirical Evaluation from Tom Sawyer							
Word	Freq. (f)	Rank (r)	$f \cdot r$	Word	Freq. (f)	Rank (r)	$f \cdot r$
the	3332	1	3332	turned	51	200	10200
and	2972	2	5944	you'll	30	300	9000
a	1775	3	5235	name	21	400	8400
he	877	10	8770	comes	16	500	8000
but	410	20	8400	group	13	600	7800
be	294	30	8820	lead	11	700	7700
there	222	40	8880	friends	10	800	8000
one	172	50	8600	begin	9	900	8100
about	158	60	9480	family	8	1000	8000
more	138	70	9660	brushed	4	2000	8000
never	124	80	9920	sins	2	3000	6000
Oh	116	90	10440	Could	2	4000	8000
two	104	100	10400	Applausive	1	8000	8000

Pawan Goyal (IIT Kharagpur) Empirical Laws Week 1: Lec 3



So, now going back to the example of Tom Sawyer, let us take some words their frequency and their rank and see whether the relation that  $f \cdot r$  is roughly constant holds to it. What you are seeing here, we have arrays words like the, and, he, there and so on, we have written their frequencies and their ranks and what is  $f \cdot r$ . So, what we are seeing here  $f \cdot r$  if you start from the fourth word say he, this remains roughly in the same range, it starts, it remains in 8000 to some 10400 and 200, the range does not vary a lot.

So, that is very nice empirical evidence that Zipf's law holds and this we will see in various corpora, that  $f \cdot r$  remains constant from most of the words that you will see.

(Refer Slide Time: 24:32)

**Zipf's Other Laws**

*Correlation: Number of meanings and word frequency*

The number of meanings  $m$  of a word obeys the law:

$$m \propto \sqrt{f}$$

Given the First law

$$m \propto \frac{1}{\sqrt{r}}$$

**Empirical Support**

- Rank  $\approx 10000$ , average 2.1 meanings
- Rank  $\approx 5000$ , average 3 meanings
- Rank  $\approx 2000$ , average 4.6 meanings

Pawan Goyal (IIT Kharagpur)      Empirical Laws      Week 1: Lec 3

So now, Zipf's law is one of the very very important law, that we have just seen, but Zipf has given some other laws also there are not so popular, but still we will try to have a look quick look at this. So, one of the laws is relating the number of meanings a word has with its frequency. So, what this law says, the number of meanings  $m$  that a word has obeys the simple law, that  $m$  is proportion to the square of roots it is frequency ( . ) .

$$m \propto \sqrt{f}$$

So, that is as we have words with higher and higher frequencies, the number of the different senses or the meanings will which they can be used also increases. So now if I use it with the first law, what do we get?

(Refer Slide Time: 25:25)

The image shows handwritten notes on a light blue background. At the top left, there is a small logo in the top right corner that reads "© CET I.I.T KGP". Below the logo, the first law is written as  $f \propto \frac{1}{r}$  with the note "- Law 1" next to it. To the right of this, the second law is written as  $m \propto \sqrt{f}$  with the note " $\rightarrow$  Law 2". Below these, a third equation is shown as  $m \propto \frac{1}{\sqrt{r}}$ . To the right of this equation, there is a note "bad part" with an arrow pointing to the term  $\frac{1}{\sqrt{r}}$ . Below the equations, there is a large bracket spanning several lines containing the text "Most words and 50% of the vocabulary occur only once." To the left of this bracket, there is a note "Most words and 50% of the vocabulary occur only once." and below it, another note "Common words are very common." To the right of the bracket, there is a note "types" and below it, another note "1/100 words account for 50% of the tokens." with an arrow pointing to the term  $\frac{1}{\sqrt{r}}$ .

So, the first law says that frequency is inversely proportional to the rank , this is law

$$(f \propto 1/r)$$

and the second law says the meanings are directly proportional to the square root of frequency ( , this is law 2. If we combine these 2 laws together, we can get meaning  
 $m \propto \sqrt{f}$ )

r inversely proportional to the rank of a word ( )(Refer Time: 25:50); so that means,

$$m \propto 1/\sqrt{r}$$

if I put the words in the decreasing order of their rank. So, starting from rank 1 to rank 2 and so on. So, the number of meanings will also keep on decreasing.

So, these clause hold an average, what is the empirical support? So as I was saying the hold on an average, if we saw the words that are having rank roughly close to 10000. So, they have roughly 2; 2.1 meanings, if they have words that are having rank roughly 5000, they are found to have roughly 3 meanings and the words that are having rank plus 2000, they were having roughly 4.6 meanings and you can do the math and you will find out this roughly follows whatever we have seen in this law.

(Refer Slide Time: 26:42)

The slide has a blue header bar with the title "Impact of Zipf's Law". Below the header, there are two main sections: "The Good part" (in a purple box) and "The Bad part" (in a pink box).  
**The Good part:** Stopwords account for a large fraction of text, thus eliminating them greatly reduces the number of tokens in a text.  
**The Bad part:** Most words are extremely rare and thus, gathering sufficient data for meaningful statistical analysis is difficult for most words.

And there is another law by Zipf that tries to correlate the length of the word; that means, the number of characters the word has with the frequency of the word and it says the frequencies of the word is inversely proportional to its length and can you try to see that in the case of English? So, the very very long words are very very rarely used, so their frequency is very very small, but some of the very small words like some simple examples like function words, they are used very very often and that also make sense because language is used for having some efficient communication. So, you cannot have long words that are very very common then you will have to write a lot. So, the words that are very very common are generally short.

So, we talked about Zipf's law. So, for doing the processing of language what is the impact of Zipf's law? So, we said there is a good part in and there is a bad part. So, what is the good part? So, in the language we saw that, remember there were 2 important observations that we made from a Zipf's law what were those? So, one observation was that 50 percent of the vocabulary; that means, the types occur only once. So, they are very rare words and this is most words are rare words, this 50 percent of the words occur only once and what was the other observation? You said that roughly 100 words account for 50 percent of the tokens. So, that is from common words are very very common, they occur a lot. So, words are very common they (Refer Time: 29:04) account of 50 percent of your text.

So, now given these observations, what do you think is the good part, what is a bad part when it comes to an end? So this is the bad part, why is it the bad part? What it says is that, so 50

percent of my words in the vocabulary they can only once, it is not easy for me to gather some statistics to do some meaning analysis. So, they are only once that is all the knowledge I have about these words, so that's why this is called the bad part, I cannot do lot of modeling for these words. And the other part that roughly 100 words account for 50 percent of token, it is we call sometimes the good part why? Because for most of the analysis you can always remove these words and that will reduce your total number of tokens by half; so we will have to process roughly half of the text, because half of the text is simply the stop words and the function words.

So, that is what I mean here, the good part is that the stop words account for a large fraction of text, thus if I eliminate the stop words it reduces the size of my text roughly by half and the bad part is most of the words are extremely rare and it is very difficult to gather data to do some meaningful analysis with each word.

(Refer Slide Time: 30:33)

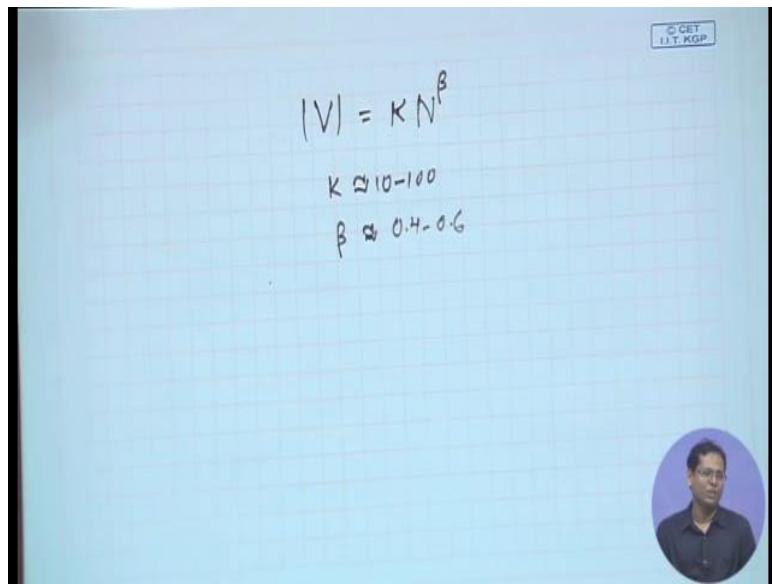
The slide has a purple header bar with the title "Vocabulary Growth". The main content area is white with a purple sidebar on the left. The sidebar contains the text: "How does the size of the overall vocabulary (number of unique words) grow with the size of the corpus?" Below this, a section titled "Heaps' Law" is shown. It states: "Let  $|V|$  be the size of vocabulary and  $N$  be the number of tokens." A mathematical equation  $|V| = KN^{\beta}$  is displayed. Below the equation, under the heading "Typically", there are two bullet points: 

- $K \approx 10-100$
- $\beta \approx 0.4_{\circ} - 0.6$  (roughly square root)

 At the bottom right of the slide, there is a circular profile picture of a man and some navigation icons.

So, now we will talk about one more law here. So, that is the relation between the size of my vocabulary and the number of tokens in my text, so that is how does the size of my vocabulary grow with the size of my corpus. So, we talked about type token ratio, so that is one sort of analysis that we can do, but that does not give me a particular law. So, what will be the rough distribution or relationship between them.

(Refer Slide Time: 31:27)



So, we have Heap's law they tries to give the relationship between the size of my vocabulary that we call as V and the number of tokens N; what this law says that it gives a symbol relationship, it says the size of my vocabulary is some constant times K, N is a number of tokens to the power beta ( ) and the ranges are also observing the corpus, so K

$$|V| = K \cdot N^\beta$$

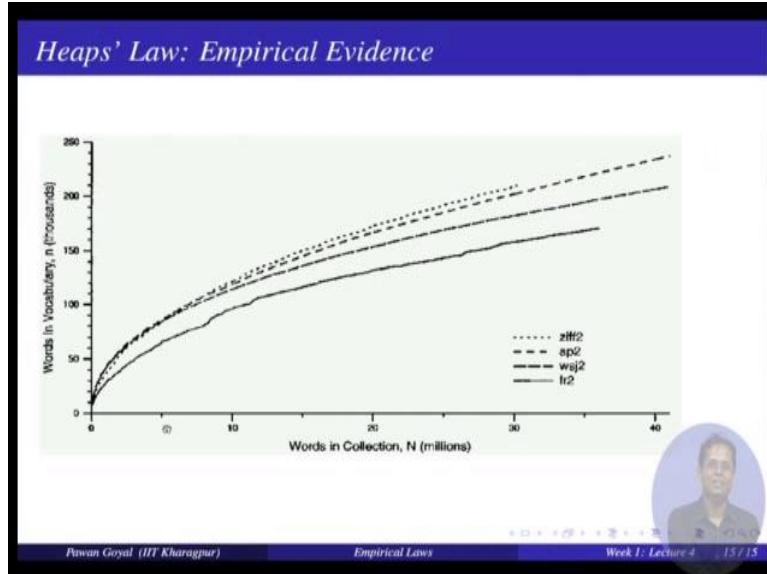
has been found to be roughly in the range of 10 to 100 ( ), although this can  $K \approx 10 - 100$

vary this is our roughly this has been found to be in this range and beta is very close to the square root ( ). So, it is found in the range of 0.4 to 0.6.

$$\beta \approx 0.4 - 0.6$$

So, what it says, vocabulary generally grows as per the square root of the number of tokens. So, that also make sense remember we are talking about what happens if I keep on increasing my vocabulary, I will tend to use the sorry, keep on increasing my tokens, I will tend to use the same word again and again. So, number of unique tokens or unique types increase roughly to the square root of my corpus size.

(Refer Slide Time: 32:22)



So, what is some sort of empirical evidence for that? So here, there are 4 different corpus that has been taken. So, like wall street journal (Refer Time: 32:30) wsj and there are some other different huge corpus and what this plot shows on the x axis, you have the words in collection in millions. So, how many words in my collection? So, there are 10 million, 20 million, 30 million, 40 million. On the y axis it is showing the number of words in a vocabulary in thousand, so 50000, 100000, 150000 and so on. And the plot shows the relationship between the vocabulary and the number of tokens and you see that the plot shows roughly a square root behavior and this has been found to match with whatever he proposed. So, this is again an empirical law that is very very universal to the language.

So, that's it for this lecture and in the next lecture we will start talking about some actual task related to the preprocessing of (Refer Time: 33:28) the corpus.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 05**  
**Text Processing: Basics**

Hello everyone. Welcome back to the final lecture of the first week. In the last lecture we were discussing about various empirical laws, in particular Zipf's law and Heap's law; that how the, what is the vocabulary are distributed in a corpus.

We saw that the distribution is not very uniform. There are certain words that are very very common. So, we saw that roughly hundred words in the vocabulary made for 50 percent of the corpus that by the time mean that number of tokens. And on the other hand there are 50 percent were words in the vocabulary that occur only once. And we discussed whatever various relationships among my vocabulary size and the number of tokens that I observe in a corpus. And also how they grow with respect to each other, and zipfs law gave me a relation between the frequency and the rank of a word.

So, today in this lecture we will start with the basic key processing in language. So, we will cover the basic concepts, and what are the challenges that one might face while doing the processing. So, we are going to the Basics of Text Processing.

(Refer Slide Time: 01:36)

The screenshot shows a presentation slide with a blue header bar containing the title "Text processing: tokenization". Below the header, there are two text boxes. The first text box is purple and contains the question "What is Tokenization?". The second text box is pink and contains the text "Depending on the application in hand, you might have to perform sentence segmentation as well.". At the bottom of the slide, there is a footer bar with the names "Pawan Goyal (IIT Kharagpur)" and "Text Processing: Basics", the week information "Week 1: Lecture 5", and the slide number "2 / 26".

So, we will start with the problem of Tokenization, as the name would suggest. Remember the name token: token is an individual word in my corpus. So now what happens when I am preprocessing the text in given in any language? What I will face is a string of characters; the sequence of characters. Now I need to identify what are all the different words that are there in this sequence. Now tokenization is a process by which I convert the string of characters into sequence of various words.

So, I am trying to segment it by the various words that I am observing. Now, before going into what is tokenization I will just talk about slight little problem sentence segmentation. So, this you may or may not have to do always and it depends on what is your application. For example, suppose you are doing classification for the whole document in to certain classes you might not have to go to the individual sentence and you can just talk about what are the various words that are present in this document.

On the other hand, suppose you are trying to find out what are the important sentences in this document; in that application you will have to go to the individual sentence. So now, if you have to go to the individual sentence the first task that you will face is how do I segment these whole documents into a sequence of sentences. So, this is sentence one, sentence two and so on, and this task is called Sentence Segmentation.

(Refer Slide Time: 03:22)

The slide has a blue header bar with the title "Sentence Segmentation". Below the header, the text "The problem of deciding where the sentences begin and end." is displayed. A section titled "Challenges Involved" contains a bulleted list:

- While '!', '?' are quite unambiguous
- Period "." is quite ambiguous and can be used additionally for
  - Abbreviations (Dr., Mr., m.p.h.)
  - Numbers (2.4%, 4.3)

A section titled "Approach: build a binary classifier" contains the text "For each ":"":

- Decides EndOfSentence/NotEndOfSentence
- Classifiers can be: hand-written rules, regular expressions, or machine learning

At the bottom of the slide, there is footer text: "Pawan Goyal (IIT Kharagpur)", "Text Processing: Basics", "Week 1: Lecture 5", and "3 / 26".

Now, you might feel that this is very trivial task, but let us see is it trivial. So what is sentence segmentation? It is a problem of deciding where my sentence begins and ends

so that I have a complete unit of words that I call as a sentence. Now do you think there might be certain challenges involved? Suppose I am talking about the language English, can I always say that wherever I have a dot it is end of the sentence? Let us see.

So, there are many ways in which I can end my sentence. So, I can have exclamation or question mark that ends the sentence and they are mostly unambiguous. So, whenever I have exclamation or question mark I can say probably this is the end of the sentence, but is the case the same with a dot. So, I can think of a scenario where I have a dot in English but it is not the end of the sentence. So, we can find all sorts of abbreviations. They end with a period like, doctor, mister, mph; so you have three dots here. So, you cannot each of this as the end of your sentence.

So, again you have numbers: 2.4, 4.3 and so on. That means the problem of deciding whether a particular dot is the end of the sentence or not is not entirely trivial. So, I need to build certain algorithm for finding out is it my end of the sentence. In text processing we face this kind of problem in nearly every simple task that we are doing. So, even if it looks a trivial task we face with this problem that can I always call dot as end of the sentence.

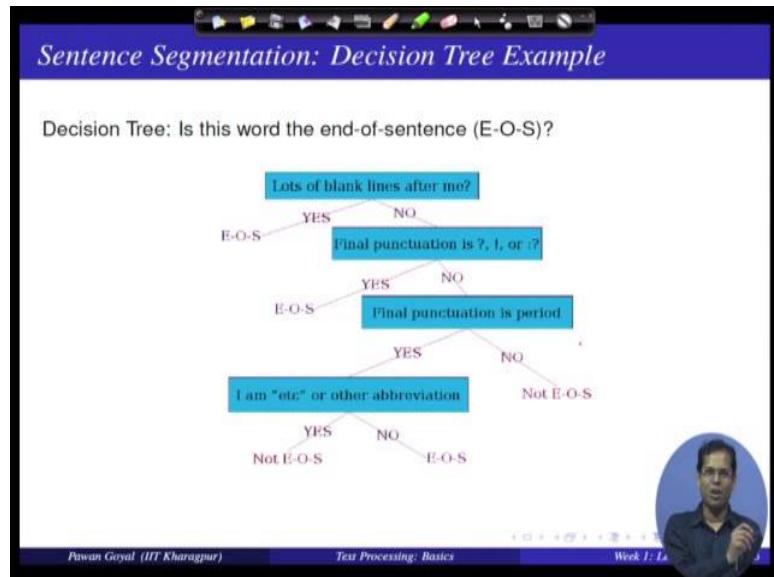
So, how do we go about solving this? Now if you think about it, whenever I see a dot or question mark or exclamation, I always have to decide one of the two things: is it the end of the sentence or is not the end of the sentence. Any data point that I am seeing I have to divide into one of these two classes. If you think of these as two classes end of the sentence or not end of the sentence, each point you have to divide into one of the two classes. And this in general, this problem in general is called classification problem. You are classifying into one of the two classes.

Now, so the idea is very very simple. So, you have two classes and each data point you have to divide into one of the two classes; that means, you have to build some sort of plural algorithm for doing that. In this case I have to build the binary classifier, what they mean by a binary classifier? There are two classes: end of the sentence or not end of the sentence. In general there can be multiple classes.

So now, for each dot or in general for every word I need to decide whether this is the end of the sentence or not the end of the sentence. So, in general my classifiers that I will build can be some rules that I write by hand, some simple (Refer Time: 06:27) nice rules

or it can be some expressions. I say my particular example matches with this set of expressions it is one class, if I does not match it is of other class. Or I can build a machine learning classifier. So, in this particular scenario what can be the simplest thing to do? Let us see. Can we build a simple rule based classifier?

(Refer Slide Time: 06:49)



So, we will start with example of a simple decision tree. So, by decision tree I mean a set of if-then-else say statements. So, I am at a particular word I want to decide whether this is the end of the sentence or not. So, I can have the simple if-then-else kind of decision tree here. So, met a word and the first thing I check is are there lots of blank lines after that. So, this would happen in a text whenever this is the end of the paragraph and there are some blank lines.

So, if I feel that there are a lot of blank lines after me; that means, after this word I may say this might be the end of the sentence with a good confidence. So, that is why the branch here says yes this is the end of the sentence. But suppose there are not lots of blank lines then I will check if the final punctuation is a question mark exclamation or a colon in that case. So, there are quite unambiguous and I may say this is the end of the sentence.

Now suppose it is not, then I will check if the final punctuation is a period. So, if it is not a period this is easy to say that this is not the end of the sentence, but suppose this is an end of this is a period. So, again I cannot say for certain if it is the end of the sentence, so

I give a again check. For simplicity I might have a list of abbreviations, and I can check if the word that I am correctly facing is one of the abbreviations in my list. If it is there I say this is not end of the sentence, if it is so here I am etcetera or any other abbreviation if the answer is yes I am not end of the sentence, if the answer is no that means this word is not an abbreviation and this will be the end of the sentence. This is very very simple if-then-else rules.

This may not be correct, but this is one particular way in which this problem can be solved. In general, you might want to use some other sort of indications; we call them as various features. These are various observations that you make from your corpus. So, what are some examples?

(Refer Slide Time: 09:02)

The slide has a blue header bar with the title "Other Important Features". The main content area contains the following bulleted list:

- Case of word with ".": Upper, Lower, Cap, Number
- Case of word after ".": Upper, Lower, Cap, Number
- Numeric Features
  - ▶ Length of word with "
  - ▶ Probability (word with "." occurs at end-of-sentence)
  - ▶ Probability (word after "." occurs at beginning-of-sentence)

At the bottom of the slide, there is footer information: "Pawan Goyal (IIT Kharagpur)", "Text Processing: Basics", "Week 1: Lecture 5", and "5 / 26".

Suppose I see the word that is ending with dot, can I use this as a feature whether my word starts with an upper case, lower case, cap, all caps or is it number. How will that help? So, let us see I am here and my word is 4.3. So, I am at dot I want to find out if it is the end of the sentence, if I can say that the current word is a number it is a high probability that this will be in number and it will not be the end of the sentence, so this can be used as another feature.

Again by feature you can think of a simple rule whether the word I am currently at is a number. Or I can use the fact where the case of the word with dots its upper case or lower case. So, what happens generally in abbreviations? We are mostly in upper case.

So, suppose I have doctor and it starts with an upper case I can say that this might be an abbreviation. Saying with the lower case: lower case will give me more probability that this is not an abbreviation.

Similarly I can also use in the case of the word after dot. So is it upper case, lower case, capital or number. So, how will that help? Again whenever I have the end of the sentence the next word in general starts with a capital. So, again this can be used. What can be some other features? So, I can have some numerical features, that is I will have certain thresholds. What is the length of the word ending with dot? Is it if the length is small it might be an abbreviation, if the length is larger it might not be an abbreviation? And I can also use probably. What is the probability that the word that is ending with dot occurs at the end of the sentence. So, if it is really the end of the sentence it might happen then that in a large corpus this end sentence quite often.

Something I can do with the next word after dot, is it the start of the sentence. What is the probability that it occurs in the start of the sentence in a large corpus? So, you might be able to use any of these features to decide given a particular word is it the end of the sentence or not. So now, suppose I ask you this question do you have the same problem in other languages like Hindi?

So, in Hindi you will see that in general there is only agenda that you use to indicate the end of the sentence and this is not used for any other purpose. So this problem you will see is again language dependent. This problem is there for English, but not so for Hindi. But you will see there are other problems that do not exist for English language, but are there for other Indian languages. We will see some of those examples in the same (Refer Time: 12:14).

(Refer Slide Time: 12:15)

The slide has a blue header bar with the title "Implementing Decision Trees". Below the header is a list of bullet points:

- Just an if-then-else statement
- Choosing the features is more important
- For numeric features, thresholds are to be picked
- With increasing features including numerical ones, difficult to set up the structure by hand
- Decision Tree structure can be learned using machine learning over a training corpus

A pink box labeled "Basic Idea" contains the following text:

Usually works top-down, by choosing a variable at each step that best splits the set of items.  
Popular algorithms: ID3, C4.5, CART

At the bottom of the slide, there is footer text: "Pawan Goyal (IIT Kharagpur)", "Text Processing: Basics", "Week 1: Lecture 5", and "6 / 26".

So, how do we implement a decision tree? As you have seen this is simple if-then-else statement. So now what is important is that you choose the correct set of features. So, how do you go about choosing the set of features? You will see in your from your data what are some observations that can separate my two classes here. So, my two classes here are; end of the sentence and non end of the sentence. And what are the observations we were having? In general it might be an abbreviation in the case of the word and that is before the dot: maybe upper case or lower case and one of these might indicate one class the other might indicate other class. So, all these are my observations that I use as my features.

Now, whenever I am using numerical features like the length of the word before dot, I need to pick some sort of threshold; that is whether the length of the word is between 2 to 3 or say more than 3 between 5 to 7 like that. So, my tree can be if the length of the word is between 5 to 7 I could one class otherwise I could another class.

Now here is one problem; suppose I keep on increasing my features it can be both numerical and non numerical features. It might be difficult to set up my if-then-else rules by hand. So, in that scenario I can try to use some sort of machine learning technique to learn this decision tree. In the literature there are lots of such algorithms available that given a data and a set of features we will construct a decision tree for you.

So, I will just give you the names of some of the algorithms. And the basic idea on this they work is that at every point we have to choose a particular split. So, you have to choose a feature value that it splits my data into certain parts. And I have certain criteria to find out what is the best way to split. So, one particular criterion is what is the information given by this. So, these algorithms that we have mentioned here like ID3, C4.5, CART they all use one of these criterions.

In general once you have identified what are your interesting features for these tasks. You are not limited to only one classifier a decision tree, you can also try out some other classifiers like.

(Refer Slide Time: 14:53)

The questions in the decision tree can be thought of as features, that could be exploited by any other classifier:

- Support Vector Machines
- Logistic regression
- Neural Networks

Pawan Goyal (IIT Kharagpur)    Text Processing: Basics    Week 1: Lecture 5    7 / 26

Support vector machines, logistic regression and neural networks. These all these are quite popular classifiers for various analytic applications. So, we will talk about some of these as we will go to some advanced topics in this course

(Refer Slide Time: 15:11)

The screenshot shows a presentation slide titled "Word Tokenization". The slide content is organized into three main sections: "What is Tokenization?", "Word Token", and "Word Type".

- What is Tokenization?**

Tokenization is the process of segmenting a string of characters into words.  
*I have a can opener; but I can't open these cans.*
- Word Token**
  - An occurrence of a word
  - For the above sentence, 11 word tokens.
- Word Type**
  - A different realization of a word
  - For the above sentence, 10 word types.

At the bottom of the slide, there is footer text: "Pawan Goyal (IIT Kharagpur)", "Text Processing: Basics", "Week 1: Lecture 5", and "8 / 26".

Now coming back to our problem tokenization; we said that tokenization is a process of segmenting a string of characters into words, finding out what are the different words in this question. Now remember we talked about token and type distinction; suppose I give you a simple sentence here I have a can opener, but I cannot open these cans.

How many tokens are there? If you count there are 11 different occurrences of words. So, you have 11 word tokens, but how many unique words are there. So, you will find there are only 10 unique words, which word repeats, is the word I repeats twice. So, there are 10 types and 11 tokens. So, my tokenization is to find out each of the 11 word tokens from the sentence.

(Refer Slide Time: 16:04)

The slide has a blue header bar with the title 'Tokenization in practice'. The main content area contains a bulleted list of tools:

- NLTK Toolkit (Python)
- Stanford CoreNLP (Java)
- Unix Commands

At the bottom, there is footer text: 'Pawan Goyal (IIT Kharagpur)', 'Text Processing: Basics', 'Week 1: Lecture 5', and '9 / 26'.

In practice at least for English you can use certain toolkits that are available like NLTK in Python, CoreNLP in Java and you can also use the Unix commands. So, in this course you will mainly be using NLTK toolkit for doing all pre processing task and in some other tasks as well, but in general you can use any of these three possibilities.

(Refer Slide Time: 16:34)

The slide has a blue header bar with the title 'Word Tokenization'. Below it, a purple box contains the heading 'Issues in Tokenization' followed by a bulleted list:

- Finland's → Finland Finlands Finland's ?
- What're, I'm, shouldn't → What are, I am, should not ?
- San Francisco → one token or two?
- m.p.h. → ??

Below this, a pink box contains the text: 'For information retrieval, use the same convention for documents and queries'. In the bottom right corner, there is a circular video player showing a person speaking.

At the bottom, there is footer text: 'Pawan Goyal (IIT Kharagpur)', 'Text Processing: Basics', 'Week 1: Lecture 5', and '9 / 26'.

So for English most of the problems that we will see are taken care of the tokenizers that we have discussed previously but, still it is good to know; what are the challenges that are involved when I tried to design a tokenization algorithm.

See for example, here you will see that if I encounter a word like Finland's in my data. So, one question that I have is whether I treat it as simple Finland, as it is Finland's or I convert it to Finland's by removing the apostrophe. So, this question you might also try to defer to the next processing step that you will see, but sometimes you might want to tackle this in the same (Refer Time: 17:14). Similarly, if you see what are, do I treat it as a single token or two tokens what are? This trouble you might have to solve in the same step, whether I treat it as a single token or multiple tokens; same with I am, should not and so on.

Similarly whenever your name end at each like San Francisco, should I treat it as a single token or two separate tokens? Now remember when we were talking about some of the cases why (Refer Time: 17:45) hard. So, you might have to find out that this particular sequence of tokens is a single entity and treat it as a single entity, not as multiple different tokens. So, this problem is related. Similarly if you find m dot p dot h whether you call it a single token or multiple tokens.

So now, there are no fixed answers to these and some of these might depend on what is the application for which you are doing this pre processing. But one thing you can always keep in mind; suppose you are doing for the application of information trivial if the same sort of steps that you apply for your documents should be applied to your query as well otherwise you will not be able to match them perfectly. Suppose, if I am using it for information trivial, so I should use the same convention for both my documents as well as the queries.

(Refer Slide Time: 18:39)

**Handling Hyphenation**

Hyphens can be

**End-of-Line Hyphen**

Used for splitting whole words into part for text justification.  
*This paper describes MIMIC, an adaptive mixed initiative spoken dialogue system that provides movie show-time information.*

**Lexical Hyphen**

Certain prefixes are often written hyphenated, e.g. co-, pre-, meta-, multi-, etc.

**Sententially Determined Hyphenation**

Mainly to prevent incorrect parsing of the phrase. Some possible usages:

- Noun modified by an 'ed'-verb: *case-based, hand-delivered*
- Entire expression as a modifier in a noun group: *three-to-five-year direct marketing plan*

Pawan Goyal (IIT Kharagpur)    Text Processing: Basics    Week 1: Lecture 5    11 / 26

So then another problem can be; how do I handle hyphens in my day? This looks again a simple problem, but we will see it is not that simple. So, let us see some kind of examples: what are the various sorts of hyphens that can be there in my corpus?

So, here I have a sentence from a research paper abstract and the sentence says this paper describes MIMIC an adaptive mixed initiative spoken dialogue system that provides movie show-time information. So, in this sentence itself you see two different hyphens: one is with initiative another is show hyphen time.

So, now can you see that these two are different hyphens; the first hyphen is not in general that I will use in my text, second hyphen I can use in my text I can write show time with an hyphen, but how did this hyphen initiative came into the corpus. So, we have given this a title end of line hyphen. So, what happens in research papers for example, whenever you write a sentence you might have to do some sort of justification and that is where you end the line even if is not the end of this of the word. So, you will end up with in hyphen.

So now, when you are trying to pre process and when you are retrieving such kind of hyphens you might have to join these together, and you should, you have to say that this is a single word initiative and not initiative hyphen. But again this is not trivial because for show time you will not do the same; show time you might want to keep it as it is.

Then there are some other kinds of hyphens like; lexical hyphens. So, you might have these hyphens with various prefixes like co-, pre-, meta-, multi-, etcetera. Sometimes they are sentimentally determined hyphens also, that is they put hyphens so that it becomes easier to interpret the sentence. Like here case-based, hand-delivered etcetera are optional.

Similarly, if you see in the next sentence three-to-five-year direct mark marketing plan; three to five year can be written perfectly without keeping the hyphens, but here you are putting it so that it becomes easier to interpret that particular occurrence. Again when you are doing tokenization your problem that how do I handle all these hyphens.

(Refer Slide Time: 21:00)

*Language Specific Issues: French and German*

**French**  
l'ensemble: want to match with un ensemble

**German**  
Noun compounds are not segmented

- Lebensversicherungsgesellschaftsangestellter
- 'life insurance company employee'
- Compound splitter required for German information retrieval

Pawan Goyal (IIT Kharagpur)      Text Processing: Basics      Week 1: Lecture 5 / 2 / 26  
10:47 AM      16/1/2016

Further, there are various issues that you might face for certain languages, but not others. For an example like in French if you have a token like ensemble, so you might want to match it with ensemble. So, that might be a similar problem that we are facing in English, but let us take something in German. So I have this big sentence here, but the problem is that this is not a single word. This is a compound composed of four different words and the corresponding English meaning is this one. So, you have four words in English. So, when you are putting in French they make a compound.

So, now what is the problem that you will face when you are processing the German text? And you are trying to tokenize it? So, you might want to find out what are the

individual words in this particular compound. So, you need some sort of compound split up for German. So, the problem is there for German not so much for English.

(Refer Slide Time: 22:26)

The slide has a blue header bar with the title "Language Specific Issues: Chinese and Japanese". Below the header, the text "No space between words" is followed by a Chinese sentence: 莎拉波娃现在居住在美国东南部的佛罗里达。 This is then broken down into individual words: 莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达. Sharapova now lives in US southeastern Florida. Below this, under the heading "Japanese: further complications with multiple alphabets intermingled.", is a Japanese sentence: フォーチュン500社は情報不足のため時間あた\$500K(約6,000万円). This sentence is annotated with boxes and arrows pointing to four categories: Katakana, Hiragana, Kanji, and Romaji. A small circular portrait of a man is visible in the bottom right corner of the slide.

So now, what happens if I am making a language like Chinese or Japanese? So, here is a sentence in Chinese. So, what do you see in Chinese words are written without any spaces in between. Now, when you are doing the pre processing your task is to find out what are the individual word tokens in this Chinese sentence. This problem is also difficult because in general for a given utterance of a sequence of characters there might be more than one possible ways of breaking into sequence of words and both might be perfectly valid possibilities.

So, in Chinese we will not have not have any space between words and I have to find out what are the places where I have to break these words; and this problem is called word tokenization. Same problem happens with Japanese and here for the complications because they are using four different steps like Katakana, Hiragana, Kanji and Romaji. So, these problems become a bit more severe.

(Refer Slide Time: 23:30)

satyambrūyāt priyambrūyānnabrūyātsatyamapriyamcanānṛtambrūyād-esadharmaḥsanātanah.

"One should tell the truth, one should say kind words; one should neither tell harsh truths, nor flattering lies; this is a rule for all times."

**Segmented Text:**

satyam brūyāt priyam brūyāt na brūyāt satyam apriyam priyam ca na anṛtam  
brūyāt esāḥ dharmaḥ sanātanah.

Now, the same problem is there even for Sanskrit. So, if some of you have taken a Sanskrit course in your class 8th or 10th you might be familiar with the rules of Sandians in Sanskrit language. So, that is it.

This is a simple single sentence in Sanskrit, but this is a huge this looks like a sing single word, it is not a single word. It is composed of multiple words in Sanskrit and they are combined with a Sandi relation. This stands for nice proverb in Sanskrit that translates in English as one should tell the truth, one should say kind words, one should neither tell harsh truths nor flattering lies; this is a rule for all times - this is a proverb.

And this is a single sentence that talks about this proverb, but there all the words are combined with Sandi relation. So, if we try to undo the Sandi this is what you will find at the segmented text. So, there are multiple words in this sentence they are combined to make a single, it looks like a single word.

So, this problem we saw in Chinese, Japanese and Sanskrit, but in Sanskrit the problem is slightly more complicated and why is that. So, in Japanese and in Chinese when you try to combine various words together you simply concatenate them, you put them one after another without making any changes at the boundary. It does not happen in Sanskrit when you combine two words you also make certain changes at the boundary and this is called the Sandi operation.

So, in this particular case since see here I have the word ‘bruyat’ and the word ‘na’, but when I am combining I am writing it ‘bruyanna’. So, you see here the letter ‘t’ gets changed to ‘n’ that means when I am trying to analyze the sentence, so this particular sentence in Sanskrit I need to find out not only what are the breaks, but what is the corresponding word from which this sentence you derived. So, from here to find out the actual words are bruyat lesna that gives me this ‘bruyat’. And this is very very common in Sanskrit that you are always combining words by doing a Sandi operation. So, this further complicates my problem of word tokenization or segmentation.

(Refer Slide Time: 26:14)

Longest Words	
Max	Language (non scientific) *
431	Sanskrit (Longest)
173	Greek
136	Afrikaans
85	Māori
79	German
74	Turkish
64	Icelandic
56	Hungarian
54	Spanish
49	Dutch
46	Malay
45	English
44	Romanian
42	Georgian
41	Czech
39	Bulgarian
39	Lithuanian
38	Kazakh
33	Norwegian
32	Tagalog
32	Polish
30	Serbian
30	Montenegrin
30	Italian
30	Croatian



Pawan Goyal (IIT Kharagpur)      Text Processing: Basics      Week 1: Lec 1

So this is just a list from Wikipedia what are the longest words in various languages. Then note this sentence is about the words. You see in Sanskrit the longest word is composed of 431 characters, it is a compound. And then you have Greek and Afrikaans and other languages, in English you will see that the longest word is of 45 characters is non-scientific.

(Refer Slide Time: 26:36)

The slide has a blue header bar with the title "Longest Words". The main content area contains a long Sanskrit compound word composed of 431 letters, followed by a detailed description of its components. At the bottom right is a circular portrait of a man, identified as Pawan Goyal. The footer includes the name "Pawan Goyal (IIT Kharagpur)", the course "Text Processing: Basics", and the week "Week 1: Lecture 5".

Compound word composed of 431 letters, from the Varadāmbikā Parinaya Campū by Tirumalāmba

निरन्तर-नृथकारिता-विग-न्तर-कन्वलबमन्द-सुधारस-विन्तु-सान्ध्रतर-धनाधन-वृन्द-संवेहकर-  
स्यन्दमान-मकरन्द-विन्द-वन्धुतर-माकन्द-तरु-कुल-तल्प-कल्प-मृदुल-सिकता-जाल-जटिल-  
मूल-तल-मरुवक-मिलवलधु-लघु-लय-कलित-रमणीय-पानीय-शालिका-वालिका-करार-विन्द-  
गलन्तिका-गलदेला-लवङ्ग-पाटल-घनसार-कस्तूरिकातिरौरभ-मेदुर-लघुतर-मधुर-शीतलतर-  
सलिलधारा-निराकरिष्ण-तवीय-विमल-विलोचन-मयूख-रेखापसारित-पिपासायास-पथिक-  
लोकान्

Pawan Goyal (IIT Kharagpur) Text Processing: Basics Week 1: Lecture 5

So, what is the particular word in Sanskrit that is composed of 431 letters? So, this was from the Varadambika Parinaya Campu by Tirumalamba; this is a single compound from his book.

(Refer Slide Time: 26:51)

The slide has a blue header bar with the title "Word Tokenization in Chinese or Sanskrit". It discusses the concept of word segmentation and provides a "Greedy Algorithm for Chinese" section. A "Maximum Matching (Greedy Algorithm)" box lists three steps: starting a pointer at the beginning of the string, finding the largest word in a dictionary that matches the string starting at the pointer, and moving the pointer over the word in the string. Below this is a pink box containing text about English word segmentation and examples of hashtags. The footer includes the name "Pawan Goyal (IIT Kharagpur)", the course "Text Processing: Basics", and the week "Week 1: Lecture 5".

Also called 'Word Segmentation'.

*Greedy Algorithm for Chinese*

**Maximum Matching (Greedy Algorithm)**

- Start a pointer at the beginning of the string
- Find the largest word in dictionary that matches the string starting at pointer
- Move the pointer over the word in string

*Think of the cases when word segmentation would be required for English Text.*

Finding constituent words in a compound hashtags: #ThankYouSachin, #musicmonday etc.

Pawan Goyal (IIT Kharagpur) Text Processing: Basics Week 1: Lecture 5 17 / 26

So now, when I talk about this problem of tokenization in Sanskrit or in English this problem is also called word segmentation, have a sequence of characters and you segment it to find out individual words. Now what is the simplest algorithm that you can think off? Let us take as in the case of Chinese. So, the simplest algorithm that works is a

greedy algorithm that is called maximum matching algorithm. So, whenever you are given a string you start you point to it at the beginning of the string. Now suppose that you have the dictionary and the words that you are currently seeing all should be in the dictionary.

So, you will find out what is the maximum match as per my dictionary in the string, you break there and put the pointer from at the next character and again do the same thing. So, this greedily chooses what are actual words by taking the maximum matches. And this works nicely for most of the cases.

So, this (Refer Time: 27:55), now can you think of some cases where the segmentation will also be required for the English text? In English in general we do not combine words to make a single word. We do not do that, but what is the scenario where we are doing that right now. So does, do hash tags come into mind. For example, suppose I have hash tags like Thank You Sachin, and music Monday. So, here different words are combined together without putting a boundary in between.

So, if you are given a hash tag and you have to analyze that you have to actually segment it into various words.

(Refer Slide Time: 28:30)

The screenshot shows a presentation slide with the following content:

**Text Segmentation for Sanskrit**

1

**General assumption behind the design**

Sentences from Classical Sanskrit may be generated by a regular relation  $R$  of the Kleene closure  $W^*$  of a regular set  $W$  of words over a finite alphabet  $\Sigma$ .

- $W$ : vocabulary of (inflected) words (*padas*) and
- $R$ : sandhi

**Analysis of a sentence**

A candidate sentence  $w$  is analyzed by inverting relation  $R$  to produce a finite sequence  $w_1, w_2, \dots, w_n$  of word forms, together with a proof that  
 $w \in R(w_1 \cdot w_2 \dots \cdot w_n)$ .

<sup>1</sup><http://sanskrit.inria.fr>

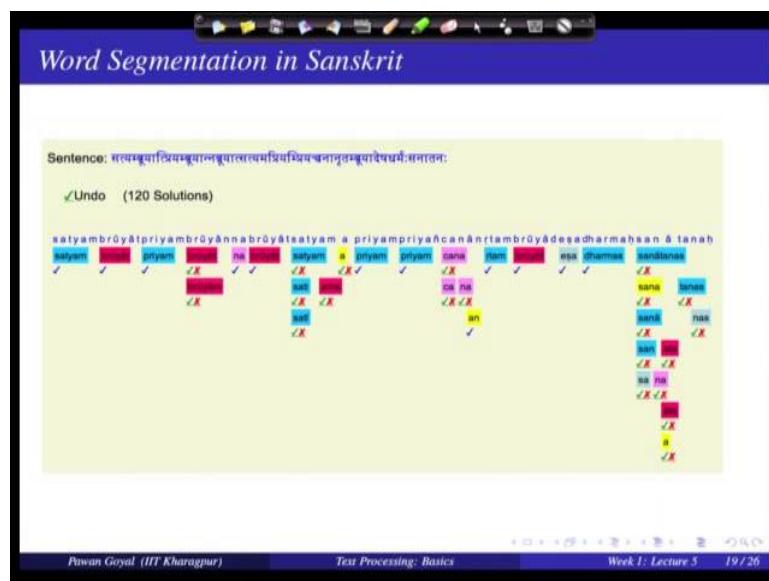
So, when I talk about Sanskrit, so this we have a segment to available at the site Sanskrit dot inria dot fr. So, we will just briefly see what is the design principle of building a

segmentor in Sanskrit? So, first we have a geometry model that says how do I generate a sentence in Sanskrit. I have a finite alphabet sigma; that means a set of various characters in Sanskrit. Now from this finite alphabet I can generate a lot words that are composed of various number of phonemes or all letters from this alphabet.

Now, when I have a set of words I can now combine them together with an operation of Sandi; that is what I mean by sigma star here; so w star here. So, I have a set of words w and I will do a cleaner closure; that means, I can combine any number of words together, but whenever I am combining words I am doing them by a Sandi operation. This is the relation between the words.

So, I have my set of inflected words also called Padas in Sanskrit and I have the relation of Sandi between them and that is how I generate sentences. But the problem is how do I analyze them. So, that is the inverse problem. That is, whenever I am given a sentence w I have to analyze it by inverting the relations of Sandi so that I can produce a finite set of word forms w<sub>1</sub> to w<sub>n</sub>. And I am saying together with the proofs so that is a formal way of saying that, but what I mean is that w<sub>1</sub> to w<sub>n</sub> whenever they combine by Sandi operation they give me the actual Sandi the initial Sandi's. So, that is how the segment is built.

(Refer Slide Time: 30:14)



Now this is a snapshot from the segmentor. So, I gave the same sentence there and it gave me all the possible ways of analyzing the Sandi's. And it says that there are 120

different solutions. So, here whenever I have bruyana, so you see there are two possibilities bruyat and bruyam. That is like that it gives me all the possible ways in which this sentence can be broken into individual word tokens.

Now this is another problem that I will have to find out what is the most likely word sequence among all these 120 possibilities. But we can use many many different models that we will not talk about in this lecture probably in some other lectures.

(Refer Slide Time: 30:52)

*Normalization*

**Why to “normalize”?**

- Indexed text and query terms must have the same form.
- U.S.A. and USA should be matched
- We implicitly define equivalence classes of terms

Pawan Goyal (IIT Kharagpur)    Text Processing: Basics    Week 1: Lecture 5    20 / 26

So coming back to normalization; we talked about this problem that the same word might be doing multiple different ways like U dot S dot A versus USA. Now I should be able to match them together. Especially, if you are doing information retrieval we are giving a query and you are retrieving from some document. Suppose your query contains U dot S dot A if the document contains USA, if you are only doing the surface able match you will not be able to map on to each other. So you will have to consider this problem in advance and do the pre processing accordingly of either your documents or the query, but using the same sort same sentence.

So, what we are doing by this? We are defining some sort of equivalence classes. We are saying USA and U dot S dot A should go to one class, and the other same type.

(Refer Slide Time: 31:52)

The screenshot shows a presentation slide with a blue header bar containing the title 'Case Folding'. The main content area contains a bulleted list of points about case folding:

- Reduce all letters to lower case
- Possible exceptions (Task dependent):
  - ▶ Upper case in mid sentence, may point to named entities (e.g. General Motors)
  - ▶ For MT and information extraction, some cases might be helpful (US vs. us)

At the bottom right of the slide is a circular video player showing a man speaking. The video player has a blue border and a small play button icon. Below the video player, the text 'Week 1: Lec' is visible.

At the bottom of the slide, there is footer text: 'Pawan Goyal (IIT Kharagpur)', 'Text Processing: Basics', and 'Week 1: Lec'.

We also do some sort of case folding that is we can reduce all the letters to lower case. So, whenever I have the word like w o r d I will always write small w o r d, so that whenever even if it is starting the sentence and it occurs in capitals because of that in general I know that this is a word w o r d. But this is not a generic rule sometimes depending on application you might have certain exceptions. For example, you might want to treat the name and it is separately. So, if you have entity General Motors you might want to keep it as it is without case folding.

Similarly, you might want to keep US for United States in upper case and not do the case folding. And this is important for the application of machine transition also, because if you do a case folding here you will know u s in lower case that means something else versus US that is in United States; excuse me.

(Refer Slide Time: 32:58)

The slide has a blue header bar with the title 'Lemmatization'. The main content area contains two bullet points:

- Reduce inflections or variant forms to base form:
  - ▶ am, are, is → be
  - ▶ car, cars, car's, cars' → car
- Have to find the correct dictionary headword form

At the bottom right, there is a circular video player showing a man speaking. The footer of the slide includes the text 'Pawan Goyal (IIT Kharagpur)', 'Text Processing: Basics', and 'Week 1: Lec 1'.

We also have the problem of lemmatization; that is you have individual words like am, are, is; and you want to convert them to their lemma; that means, what is the base form from which they are derived. Similarly car, cars, car's cars'; so all these are derived from car. Again this is some sort of normalization we are saying all these are some sort of equivalence class because they come from the same word from.

So, in the problem of lemmatization is that you have to find out the actual dictionary head word from which they have derived.

(Refer Slide Time: 33:32)

The slide has a blue header bar with the title 'Morphology'. The main content area contains a statement: "Morphology studies the internal structure of words, how words are built up from smaller meaningful units called **morphemes**". Below this, a purple box contains the text: "Morphemes are divided into two categories". A list follows:

- Stems: The core meaning bearing units
- Affixes: Bits and pieces adhering to stems to change their meanings and grammatical functions
  - ▶ Prefix: un-, anti-, etc (a-, ati-, pra- etc.)
  - ▶ Suffix: -ity, -ation, etc (-taa, -ke, -ka etc.)
  - ▶ Infix: 'n' in 'vindati' (he knows), as contrasted with *vid* (to know).

At the bottom right, there is a circular video player showing a man speaking. The footer of the slide includes the text 'Pawan Goyal (IIT Kharagpur)', 'Text Processing: Basics', and 'Week 1: Lec 1'.

And for that we use morphology. So, what is morphology? I am trying to find out the structure of word by seeing what is the particular stem the headword and what is the affix that is applied to it. So, these individual units are called various morphemes.

So, you have stems that are the (Refer Time: 33:55) hybrids and the affixes that are what are the different units like as for plural etcetera you are applying to them to make the individual word. So, my examples are like for prefix you have un, anti, etcetera for English and a-, ati-, pra- etcetera for Hindi or Sanskrit. Suffix like ity, -ation etcetera and -taa -ka -ke etcetera for Hindi. And in general you can also have some infix, like you have the word like vid and you can infix n in between this is in Sanskrit. So, we will discuss in detail about it in morphology later.

So, there is another concept you have lemmatization where you are finding the actual dictionary headword. So, there is also a concept called stemming where you do not try to find the actual dictionary headword, but you just try to remove certain suffixes and whatever you obtain is called a stem, so this crude chopping of various affixes in that word.

(Refer Slide Time: 34:59)

The screenshot shows a presentation slide with a blue header bar containing the title 'Stemming'. Below the title is a list of bullet points:

- Reducing terms to their stems, used in information retrieval
- Crude chopping of affixes
  - ▶ language dependent
  - ▶ *automate(s), automatic, automation* all reduced to *automat*

Two red callout boxes with arrows point from the text below the list to the right side of the slide. The left box contains the text: "for example compressed and compression are both accepted as equivalent to compress.". The right box contains the text: "for example compress and compress are both accepted as equivalent to compress".

At the bottom of the slide, there is a footer bar with the text "Pawan Goyal (IIT Kharagpur)", "Text Processing: Basics", "Week 1: Lemmatization", and a small circular profile picture of a person. The slide is displayed on a computer screen with a taskbar visible at the very bottom.

So, this is again language dependent. So, what we are doing here words like automate, automatic, automation all will be reduced to a single lemma automatically. So, this is stemming, so you know the actual lemma is automate with an e, but here I am just

chopping off the affixes at the end. So, I am removing here this ic, ion all and putting it to automate.

So, this is one example; if you try to do a stemming here see you will find from example e is removed, from compressed ed is removed and so on. So, what is the algorithm that is used for this stemming?

(Refer Slide Time: 35:48)

Porter's algorithm

**Step 1a**

- sses → ss (caresses → caress)
- ies → i (ponies → poni)
- ss → ss (caress → caress)
- s → φ (cats → cat)

**Step 1b**

- (\*v\*)ing → φ (walking → walk, king → king)
- (\*v\*)ed → φ (played → play)

Pawan Goyal (IIT Kharagpur)    Text Processing: Basics    Week 1: Lec.

So, we have the Porter's algorithm that is very very famous. And this is again some sort of if-then-else rules. So, what are some examples here? What is the first step? I take a word if it ends with sses I remove es from there and I end with ss, so example is caresses goes to caress. If not then I see whether the words end with ies I put it to i - like ponies goes to poni. If not I see if the word ends with ss I keep it as ss, if not I see if the word ends with s I remove that s. Cats goes to cat but caress does not go to caress with only one s because this is step comes before. If there is a double s and in the word I written it otherwise if there is a single s I remove it that.

Like that there are some other steps. So, if there is a vowel in my word and the word ends with ing I remove ing. So, walking goes to walk, but what about king you see in k there is no vowel. So, king will be written as it is. Same is a vowel and there is an ed I remove this ed. And I have this word played to play. So, you can see that what is the use of this heuristic of having this vowel. If you did not have this vowel you would have converted king to k.

(Refer Slide Time: 37:17)

*Porter's algorithm*

*Step 2*

- ational → ate (relational → relate)
- izer → ize (digitizer → digitize)
- ator → ate (operator → operate)

*Step 3*

- al → φ (revival → reviv)
- able → φ (adjustable → adjust)
- ate → φ (activate → activ)

Pawan Goyal (IIT Kharagpur)    Text Processing: Basics    Week 1: Lec 1

And like that there are some other ways like if the word ends with ational then I will put it put ate, so rational; so relational to relate. And if the word ends with izer I convert I remove that r digitizer to digitize ator to ate. And if the word ends with al I remove that al, if the word ends with able I remove that able, if the word ends with ate I remove that ate. So, like that these are some steps that I take from my corpus from each word I convert it to its step, it does not give me the correct dictionary headword, but still this is a good practice in principle for information retrieval, if you want to match the query with the documents.

This is for this week. Next week we will start with another pre processing task that is a spelling correction.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute Technology, Kharagpur**

**Lecture - 06**  
**Spelling Correction: Edit Distance**

Welcome back to the second week of this course. So, last week we handed with some the concept in pre-processing of the text, so what are the concepts that we covered? We covered given is sentence or document in general; a text document, how do I cyber into a sentences, how do I segment into words; we saw what are specific issues that might arise because of various languages and we talked about normalization, case folding and other aspects like lemmatization and stemming.

So these are some very very standard pre-processing tasks that you might have to do on a given text in a given language. But there is another task that you might have do, in certain cases, if the data that you are obtaining is bit noisy; what I mean noisy? There are certain spelling errors and you want to correct them for doing your analysis. So this task is called spelling correction. So, what are the various problems that involved in spelling correction and what are the different algorism that we will be using?

So, we will be start with the very very simple algorithm of how do you use at the distance for spelling correction in this particular lecture.

(Refer Slide Time: 01:42)

The screenshot shows a presentation slide with the following content:

- Section Header:** Spelling Correction
- Text:** I am writing this email on behalf of ...  
The user typed 'behaf'.
- List:** Which are some close words?
  - behalf
  - behave
  - ....
- Section Header:** Isolated word error correction
- List:**
  - Pick the one that is closest to 'behaf'
  - How to define 'closest'?
  - Need a **distance metric**
  - The simplest metric: **edit distance**
- Page Footer:** Pawan Goyal (IIT Kharagpur) | Spelling Correction: Edit Distance | Week 2: Lecture 1 | 2 / 20

So, what is the problem of start a spelling correction; suppose you are reading or you are seeing this text in your data and the sentence is I am writing on behalf of, but in behalf you see only b e h a f and l is missing. So, now you want to correct this particular word to the actual word behalf, now what is the algorithm approach of solving this problem? So, in general if you see this word like behalf, so ideally we would want to find out what are the closest words in English that might come in place of this. So, other words you can think of; I can think of probably behalf and I can think of probably behave yes and there might be some other words.

Now so when I will be talking about the spelling correction, I will talk about two different scenarios; one is isolated word error correction; that is I am trying to correct a word that is incorrect, but I am not trying to use the context around it. So, given the word b e h a f, I want to correct it without choosing what is there before and after this word and this is called isolated word error detection. So now, in this approach I will try to find the word that are closest to behalf; you see maybe behalf and behave will be two words.

Now so immediately I will crop them, so how do I define what do I mean by closest. So, ideally we say where you have not some sort of match between this word and that word, but how do I further define it. So, I need some sort of distance metric, how do I measure distance between one word to another word and as per this metric whichever two words are coming out to be closer will be called as closest, if they were coming too far I will not even consider as a candidate for the correction. So, I really want distant metric that can give me the b e h a f and b e h a l f are very very closed together. So, we will start and we will actually use the metric called edit distance and this is one of the simplest metric that we can think of in this case. Now what is edit distance?

(Refer Slide Time: 04:04)

The screenshot shows a presentation slide with a blue header bar containing icons for file operations like Open, Save, Print, etc. The main title 'Edit Distance' is centered above the content area. The content area contains a bulleted list defining edit distance:

- The minimum edit distance between two strings
- Is the minimum number of editing operations
  - ▶ Insertion
  - ▶ Deletion
  - ▶ Substitution

At the bottom of the slide, there is footer information: 'Pawan Goyal (IIT Kharagpur)', 'Spelling Correction: Edit Distance', 'Week 2: Lecture 1', and '3 / 20'. The bottom navigation bar includes standard presentation controls like back, forward, and search.

So what is my problem? In general I am given two strings; one that is incorrect another might be correct and I have to find out what is the minimum distance between these two strings, so that is what I want to define by my concept of edit distance. Now how is it defined, so minimum edit distance is defined as the minimum number of edit operations that I have to do for going from one string to another one, as a name say edit distance; how many edits you are making to go from one string to another string; that is how it is defined.

Now immediately I will have the next question; what are all the various edit operations I can make for going from one string to another string. So, we will start with three basic operations that are insertion; that means, I insert a character in the first string, deletion; that means, I delete a character in the first string or substitution; I delete or I substitute one character in the first string by another character. So, these are three different operations that we will consider. So, now my question would be how many of these operations are needed to go from one string to another string that is my edit distance between two strings.

(Refer Slide Time: 05:30)

The screenshot shows a presentation slide titled "Minimum Edit Distance". A purple box labeled "Example" contains the text "Edit distance from 'intention' to 'execution'". Below this, two words are shown with vertical lines and red annotations indicating operations: "I N T E \* N T I O N" and "\* E X E C U T I O N". Handwritten annotations show a deletion (deletion of 'I'), three substitutions ('T' to 'X', 'N' to 'E', 'O' to 'U'), and an insertion ('C'). A video player interface at the bottom shows a speaker icon and the text "Pawan Goyal (IIT Kharagpur) Spelling Correction: Edit Distance Week 2: Lecture 1".

So, let us take a simple example, so I have the initial word is intention and the final word is execution and I want to find out how many of these operations are required to go from intention to execution. So, now if I try to see that, so what am I doing to go from intention to execution; I am first doing a deletion, I deleted I. Now I substitute it N with E, again I substitute it T with X; I kept E as it is, then I inserted C.

So, here star means I went to null that is the deletion; here null went to C that is insertion. Then again I did a substitution and with U and T, I, O, N remain as it is. So, how many operations I needed? I needed five different operations; one was deletion, one insertion and three substitutions, so here the various operations that I did. Now I need to define for these operations what will be the edit distance.

(Refer Slide Time: 07:04)

The screenshot shows a presentation slide titled "Minimum Edit Distance". The slide contains two strings:

I N T E * N T I O N	
* E X E C U T I O N	
d s s      i s	

Below the strings is a bulleted list:

- If each operation has a cost of 1 (Levenshtein)
  - ▶ Distance between these is 5
- If substitution costs 2 (alternate version)
  - ▶ Distance between these is 8

At the bottom of the slide, there is a circular profile picture of a man, Pawan Goyal, and the text "Pawan Goyal (IIT Kharagpur) Spelling Correction: Edit Distance Week 2: Levenshtein Distance".

So, there are two difference variations that I can use; in one variation I might say that each operation has cost of 1. So, whether I am doing deletion intention or substitution each has a cost of 1. So, in this case what will be the edit distance between the two strings because I have done five operations; each has a cost of 1, so distance will be 5, so this is called Levenshtein distance.

Now I may also use another variant where the substitution has a cost of 2 because this is one insertion, one deletion. So, if I take substitution cost as 2; what will be the edit distance between the two strings? So, when I have 1 plus 1 for one insertion, one deletion plus 3 times 2 for 3 substitutions that will be 8.

(Refer Slide Time: 08:09)

How to find the Minimum Edit Distance?

Searching for a path (sequence of edits) from the *start string* to the *final string*:

- Initial state: the word we are transforming
- Operators: insert, delete, substitute
- Goal state: the word we are trying to get to
- Path cost: what we want to minimize: the number of edits

```
graph TD; intention[intention] -- Del --> ntention[ntention]; intention -- Ins --> eintention[eintention]; intention -- Sub --> entention[entention]
```

Pawan Goyal (IIT Kharagpur)      Spelling Correction: Edit Distance      Week 2: Lecture 1

So, now my question is that how do we find the minimum edit distance between two strings, so what will be the approach for doing metric. So let us define it formally what are we trying to achieve, so I am trying to search for a path, by a path I mean what are the sequence of edit operations that I have to make for going from the start string to my final string. So now I can define what is my initial state; that is what is the word that I am trying to transform?

So in the previous example intention, so this is my initial state then what are the various operators that I am allowed, what various operators that I am allowed to use over my over any of the strings; I am allowed to use insertion, deletion or substitution; these are my three operators and there is a one goal state that is the final string that I am trying to get. So, it will be execution in the previous case and I want to minimise the path cost; number of edits, I want to find out what is the minimum number of operations or minimum path cost by which I can go from initial strings to final string. So, now if you think of it in a very naive manner, what will be an approach that you might try? So, I am starting with intention and I have to reach execution. So, one very naive method might be you start with intention and try out all possible edit operations until unless you reach execution.

So, that is I shall intention; if I delete one character; I can delete i, I go to n intention. Similarly you might delete other characters, you might insert one character in the

beginning say e or you can insert in any of the places or you might try to substitute one character like e can be substituted by; I can be substituted with e and that gives you another word and you might keep on trying all these operations until there is a point when you are at the goal state. So, this might be one possible approach; now what is even problem that you see where is approach. So, don't you think you might have to travel a lot of paths that are not at all relevant for my problem, why should I keep deleting i n t e n t i o n all together in one path when this will never give me execution.

(Refer Slide Time: 10:44)

*Minimum Edit as Search*

*How to navigate?*

- The space of all edit sequences is huge
- Lot of distinct paths end up at the same state
- Don't have to keep track of all of them
- Keep track of the shortest path to each state

Pawan Goyal (IIT Kharagpur)      Spelling Correction: Edit Distance      Week 2: Lecture 1      7 / 20

So here I am trying to navigate all the possible edit operations to achieve the goal state, but this gets me huge space, we will try to work out one simple example that how many even if I try to find words with the addition one or two, it might be the number of operation, the number of possibility might be huge, so how can I optimise this. So, in this case what would also happen; lot of paths might end up with the same state.

So, I might end up with doing some insertion and another deletion that might end up in the same state I was starting with or it might have an insertion, deletion or another substitution. So, again this is unavoidable that kind of approach, but we will like to avoid it. Further do I really need to keep track of all of them, I need only those paths that will come between intention to execution. So the idea would be; I define some sort of the states and keep track what are shortest paths to each state because ideally I want to find it what is the shortest distance between two strings. So, there is no reason that I should

store all possible wage of arriving from one string to another that are having a higher distance because I want to keep the minimum distance path.

(Refer Slide Time: 12:21)

For two strings

- $X$  of length  $n$
- $Y$  of length  $m$

We define  $D(i,j)$

- the edit distance between  $X[1..i]$  and  $Y[1..j]$
- i.e., the first  $i$  characters of  $X$  and the first  $j$  characters of  $Y$

Thus, the edit distance between  $X$  and  $Y$  is  $D(n,m)$

Pawan Goyal (IIT Kharagpur) Spelling Correction: Edit Distance Week 2: Lecture 1 8 / 20

So what is an approach that I will use? So for that let us try to define it more formally. So, I have two strings  $X$  and  $Y$ ;  $X$  is of length  $n$  and  $Y$  is of length  $m$  and I am trying to find out the distance from  $X$  to  $Y$ , so in general I can define some edit distance metrics.

So how will I define it, so let us see my definition  $D(i,j)$ ; where an element  $D(i,j)$  denotes the edit distance between the first  $i$  characters of  $X$  ( $X[1..i]$ ) and first  $j$  characters of  $Y$  ( $Y[1..j]$ ). So, how many operations I need to make to go from  $i$  characters of  $X$  to  $j$  characters of  $Y$ . So, now as per this definition what will be the edit distance between  $X$  and  $Y$ ; this will be simply  $D(n,m)$ ; that means, the  $n$  characters of  $X$  and  $m$  characters of  $Y$ . So now, question is what kind of algorithm I should use for obtaining the element of this matrix.

(Refer Slide Time: 13:30)

The screenshot shows a presentation slide with a blue header bar containing icons. The main title is "Computing Minimum Edit Distance". Below the title, there is a purple box with the heading "Dynamic Programming". Inside this box, there is a bulleted list of steps:

- A tabular computation of  $D(n,m)$
- Solving problems by combining solutions to subproblems
- Bottom-up
  - ▶ Compute  $D(i,j)$  for small  $i,j$
  - ▶ Compute larger  $D(i,j)$  based on previously computed smaller values
  - ▶ Compute  $D(i,j)$  for all  $i$  and  $j$  till you get to  $D(n,m)$

At the bottom of the slide, there is footer information: "Pawan Goyal (IIT Kharagpur)", "Spelling Correction: Edit Distance", "Week 2: Lecture 1", and "9 / 20".

So, there we will try to see this problem as a dynamic programming approach, so we will compute  $D_{n,m}$  in a tabular manner. In general what do we do in dynamic programming, so whenever we have to solve the problem; we try to divide it in such a manner that I first solve the smaller problems and try to use their solutions to solve the bigger problem so that I can avoid certain habitations in my computations. So, I will first start by solving  $D_{i,j}$  for small values of  $i$  and  $j$  and I will implemently use that to find out finally, my  $D_{n,m}$ ; by using the smaller values that I have already computed.

So, this is the bottom of approach; I first solve the very smalls of problems and then I use it to solve the bigger problems. So, I compute  $D_{(i,j)}$  for small  $i$  and  $j$  and based on the smaller values that I have computed, I will compute larger values of  $D_{(i,j)}$  and I keep on doing it until I get  $D_{(n,m)}$ , so this will be our basic idea.

(Refer Slide Time: 14:56)

*Dynamic Programming Algorithm*

---

**Initialization:**

$$D(i, 0) = i$$

$$D(0, j) = j$$

**Recurrence Relation:**

```
For each i = 1..M
  For each j = 1..N
    D(i, j) = min {
      D(i-1, j) + 1
      D(i, j-1) + 1
      D(i-1, j-1) + 2; [if x(i) ≠ y(j)]
      0; [if x(i) = y(j)]}
```

**Termination:**

$$D(N, M) \text{ is distance}$$

*Pawan Goyal (IIT Kharagpur)*   *Spelling Correction: Edit Distance*   *Week 2: Lec 1*

So, now what we will need to see that; how should I will be using the smaller computations to get the value for the large amounts? So, I can define; so here I am I can define my dynamic programming algorithm for finding out the edit distance between two strings. So, first we are doing the initialization, so there we are saying  $D(i, 0)$  is  $i$  and  $D(0, j)$  is  $j$  now does that make sense, what is  $D(i, 0)$  that is the distance between the  $i$  characters of  $X$  and 0 characters of  $Y$ . Now what will be the distance, how many operations I have to make for going from  $i$  characters of  $X$  to 0 characters of  $Y$  and how can I do that, the only possible way of doing it efficiently is I keep on deleting each of the character. So, there will be  $i$  operations of deletions, the cost will be  $i$ . Similarly how we go from 0 character of  $X$  to  $j$  characters of  $Y$ , I can do  $j$  different intention. So, that is why I can initialize it as  $D(i, 0)$  is  $i$  and  $D(0, j)$  is  $j$ ; that is fine.

Now, in general how do I define  $D(i, j)$  in terms of the previous the other short elements that I have already filled up, so here are 3 (Refer Time: 16:13) by which  $D(i, j)$  might be filled. So, suppose I am trying to find out the edit distance between  $X$  that is having; right now  $i$  characters of  $X$  and  $Y$  that is having  $j$  characters. Now the idea is that how go I compute  $D(i, j)$  using the smaller values of  $i$  and  $j$ . So, that is how I use  $D(i-1, j)$ ;  $D(i, j-1)$  and  $D(i-1, j-1)$ , so let us see in this example. So, I want to use  $D(i-1, j)$ , so what is  $D(i-1, j)$ .

So, this is suppose one character, so this is my  $i$  minus 1; distance between  $X$   $i$  minus 1 ( $X(i-1)$ ) to  $Y$   $j$  ( $Y(j)$ ) and suppose we have already compute it, what is the minimum cost between  $X$   $i$  minus 1 characters and  $Y$   $j$  characters. Now how do I use to find out what might be the minimum possible edit distance between  $X$  and  $Y$   $j$ . So, for the  $i$  characters in  $X$ ; I can do a deletion, I go to  $X$   $i$  minus 1 and then I use the distance between  $X$   $i$  minus 1  $Y$   $j$ . So, how can I write the distance; I can say that same distance that much would be  $D$   $i$  minus 1  $j$ ; plus 1 because I deleted one character from  $X$ ; is that clear; in  $X$   $i$  I delete one character, I went to  $X$   $i$  minus 1 and I have already computed distance between  $X$   $i$  minus 1 and  $Y$   $j$ , so that is one for deletion and this cost we already have; similarly how do you use the use  $D$   $i$   $j$  minus 1 ( $D(i-1, j-1)$ ); same idea.

(Refer Slide Time: 18:19)

```

Initialization
D(i,0) = i
D(0,j) = j

Recurrence Relation:
For each i = 1..M
    For each j = 1..N
        D(i,j) = min {
            D(i-1,j) + 1
            D(i,j-1) + 1
            D(i-1,j-1) + 2; if x(i) ≠ y(j)
            0; if x(i) = y(j)
        }

Termination:
D(N,M) is distance

```

So, here I have  $X$  and  $Y$ ;  $X$  is  $i$ ;  $Y$  is  $j$  and suppose I have already computed  $i$   $j$  minus 1. So, how will I compute the distance between  $i$  and  $j$ ,  $i$  is the same distance that I computed between  $i$  and  $j$  minus 1 plus I inserted the  $j$ th character. So, this you can think of it as insertion, so distance will be  $i$   $j$  minus 1 and 1 and what might be the third possibility; third possibility would be that I know what is the distance between  $i$  minus 1  $j$  minus 1 and I am seeing whether I am substituting  $i$  with  $j$ .

So there might be two ways; if each character is the same; that means I am not having any cost, but if there are different characters, I will have a cost corresponding to the substitution. So,  $D$   $i$  minus 1  $j$  minus 1 plus 2 if the characters are not same and 0 if they

are same. So, there are three ways in which I can write  $D_{i,j}$ ; it can made  $D_{i-1,j} + 1$ ;  $D_{i-1,j+1}$ ; or  $D_{i-1,j-1}$ . If the characters are different or 0 if the character are same, so there is three different ways I can define these cost. Now the cost we are only store in the minimum cost, I will see what is the minimum of all three cost and that is what I will storage as  $D_{i,j}$  and that I keep on doing this until I arrive it  $D_{n,m}$ . So, that is the simple dynamic programming approach for solving this.

(Refer Slide Time: 20:13)

N	9								
O	8								
I	7								
T	6								
N	5								
E	4								
T	3								
N	2								
#	1								
#	0								

So, let us see once one example; now so same example intention and execution. So, we have already filled up some entries here; now can you relate to this, what is this entry this is my  $D_{1,0}$ . So, initialization we saw that  $D_{1,0}$  will be simply 1 so that means I can simply delete; I can delete this character and I arrive at the final sub history; starting from i; how do I go to null; I delete i; so this is 1.

Similarly this will be 2; I can delete i and n; one at a time, this will be 3 i n t and all so on. Similarly why this is 1 from null to e; that means, I am inserting one character, null to e X i am inserting two characters. So, this will 1, 2 and so on, so that explains all these initial Levenshtein distance in this step. Now my problem is how do I fill up the other (Refer Time: 21:25) in this table, so for example, how I fill this particular (Refer Time: 21:29).

(Refer Slide Time: 21:35)

So now, how did we define my dynamic programming approach, I said I filled  $D_{i,j}$  based on  $D_{i-1,j}$ ,  $D_{i,j-1}$  and  $D_{i-1,j-1}$ . Now in this case what is  $D_{i-1,j}$ , suppose I want to fill this element; this is  $D_{i,j}$ , what is  $D_{i-1,j}$ ; this is my  $i-1,j$ , what is my  $D_{i,j-1}$ ; this is my  $i,j-1$  and what is my  $i-1,j-1$ , this is my  $i-1,j-1$ . Now what will be the cost here; it will be minimum of all these three operations. So, let us find out for individual  $D_{i-1,j}$  is 1 plus 1, so this gives me 2;  $D_{i,j-1}$  cost between null and I am sorry I think I wrote it incorrectly, it should be  $D_{i,j-1}$  and there should be the  $D_{i-1,j-1}$ .

So, what is  $D[i-1][j]$  that is from null to  $E$ , what is the cost 1 plus 1 that will be 2;  $D[i][j-1]$  minus 1 is  $i$  to null that is 1, plus 1 this will again give me 2 and  $D[i-1][j-1]$  is  $j=0$ ; plus 2 if the two characters are not the same. So, here the two characters  $i$  and  $E$  there are not the same; so this will be 2. So, that is I can go from any of these paths and I have a cost of 2, so minimum will again by 2. So, I enter here, so now suppose you want to use that to find out the value here. So, now one thing we can see, I need three different entries this, this and this. So, what will be the cost minimum of 3, 3, 3 again this will be 3.

So, like that I will keep on filling up all the table until I arrive at this element; that is my  $D_{n,m}$ . So, in general there might be more than one ways of arriving at the minimum cost, so we will see how do we store that, but right now we can just put that cost as it is.

(Refer Slide Time: 24:28)

N	9	8	9	10	11	12	11	10	9	8
O	8	7	8	9	10	11	10	9	8	9
I	7	6	7	8	9	10	9	8	9	10
T	6	5	6	7	8	9	8	9	10	11
N	5	4	5	6	7	8	9	10	11	10
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

So we fill up this complete matrix, this is what I will get and you see the distance between intentions, execution; in this case is 8. But you also see in this slide that for certain strings, the edit distance might go up to 12 that is possible, but finally the edit distance between intention, execution comes up to be 8. So, this approach gives me very efficiently what is the edit distance between one string and another string and we know how to use this. Now so there is one further thing that we can do with this algorithm, so that is again depending on my algorithm; my particular task.

(Refer Slide Time: 25:17)

- Computing edit distance may not be sufficient for some applications
  - ▶ We often need to align characters of the two strings to each other
- We do this by keeping a "backtrace"
- Every time we enter a cell, remember where we came from
- When we reach the end,
  - ▶ Trace back the path from the upper right corner to read off the alignment

That is about computing alignments, so for some applications we just want to know what is the edit distance between two strings but for other applications, you might also want to find out what are all the places where the edit took place, so what is the alignment. Remember the way we did the case of intention versus execution, we found out what are the actual places where insertion, deletion, substitution are taking place. So, this is what I mean by alignment which parts are line to the other part in the string.

So, now the question is can I modify the same algorithm to also find out what will be the various alignments in the string, an you will see it is very very easy by use the previous algorithm that we have seen. For some application, we might need to align characters of the two strings to each other. Now we do this by keeping a sort of backtrace, so what do you mean by backtrace. So whenever I am filling out the value of  $D_{i,j}$ ;  $i$  can come off from either  $D_{i-1,j}$ ;  $D_{i,j-1}$  or  $D_{i-1,j-1}$ .

So whichever gives me the minimum cost, I will put a pointer that I came from either form left, from bottom or diagonal element. So whenever I am at  $D_{n,m}$ , I will again start doing a backtrace from there which element I came here from and I keep on doing it from started from there until I arrive at the null characters. So every time we enter a cell, we remember where we came from and when we reach the end, we trace back the path from the upper right corner to read all the alignment.

(Refer Slide Time: 27:08)

N	9									
O	8									
I	7									
T	6									
N	5									
E	4	3	4							
T	3	4	5							
N	2	3	4							
I	1	2	3							
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

Pawan Goyal (IIT Kharagpur) Spelling Correction: Edit Distance Week 2: Lec 1

(Refer Slide Time: 28:14)

*Minimum Edit with Backtrace*

n	9	↓ 8	↖-1 9	↖-1 10	↖-1 11	↖-1 12	↓ 11	↓ 10	↓ 9	↖ 8	
o	8	↓ 7	↖-1 8	↖-1 9	↖-1 10	↖-1 11	↓ 10	↓ 9	↖ 8	← 9	
i	7	↓ 6	↖-1 7	↖-1 8	↖-1 9	↖-1 10	↓ 9	↖ 8	← 9	← 10	
t	6	↓ 5	↖-1 6	↖-1 7	↖-1 8	↖-1 9	↖ 8	← 9	← 10	↖ 11	
n	5	↓ 4	↖-1 5	↖-1 6	↖-1 7	↖-1 8	↖-1 9	↖-1 10	↖-1 11	↖ 10	
e	4	↖ 3	← 4	↖-1 5	↖-1 6	↖-1 7	↖-1 8	↖-1 9	↖-1 10	↓ 9	
t	3	↖-1 4	↖-1 5	↖-1 6	↖-1 7	↖-1 8	↖ 7	↖-1 8	↖-1 9	↓ 8	
n	2	↖-1 3	↖-1 4	↖-1 5	↖-1 6	↖-1 7	↖-1 8	↓ 7	↖-1 8	↖ 7	
i	1	↖-1 2	↖-1 3	↖-1 4	↖-1 5	↖-1 6	↖-1 7	↖ 6	↖-1 7	↖-8	
#	0	1	2	3	4	5	6	7	8	9	
	#	e	x	e	c	u	t	i	o	n	

Pawan Goyal (IIT Kharagpur) Spelling Correction: Edit Distance Week 2: Lecture 1 16 / 20

So, let us see how we will do that in this case, so suppose I was filling up this element and I know I can come from either here or here or here; all three are equally likely. So, I might put all three points, but suppose when I was here the best possible way was this one. So, I should have come from the diagonal to this element, so this is what I will store, similarly here I will store another back point and so on. So, now once my edit operation is over; all you have these back point is each and every cell, I start from D n m and keep on following by back point.

So, suppose I follow this one, I reach this cell again I follow point I reach this cell this one and so on until I reach the null characters and that will give me the alignment of two strings. So you will have alignment starting from D n m, so we will see from here I can go to this element, this element, this element. Now when I am here, it could have come from either of the three elements. So, you might have some sort of preference; you might say I whenever there equal to equal possibility among all three, I will always diagonal this you might say. So, you come to this element then you go to left, then again you go down based on preference, down based on preference, down based on preference and then that is where you get your alignment. So, this is some way you can compute which characters in intention are aligned to which character in executions.

(Refer Slide Time: 29:03)

*Adding Backtrace to Minimum Edit*

```

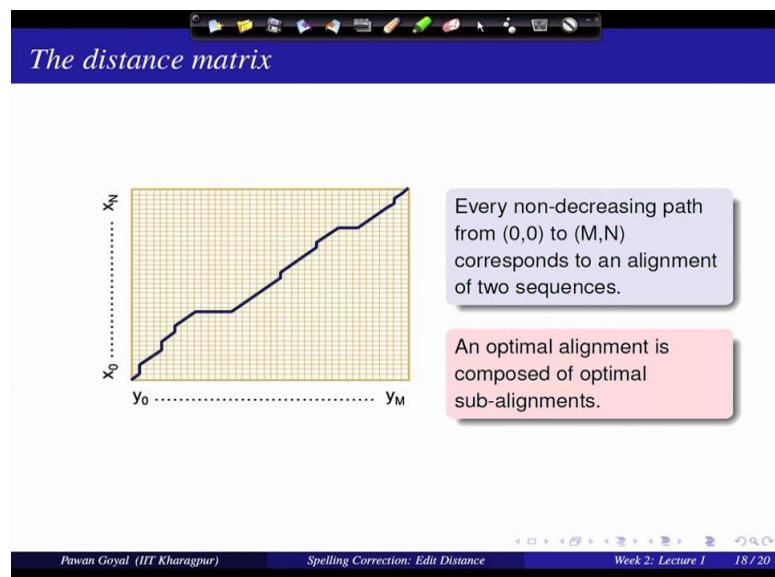
Base conditions:           Termination:
D(i,0) = i             D(0,j) = j             D(N,M) is distance
Recurrence Relation:
For each i = 1..M
    For each j = 1..N
        D(i,j) = min {
            D(i-1,j) + 1      deletion
            D(i,j-1) + 1      insertion
            D(i-1,j-1) + 2; [
                if X(i) ≠ Y(j) substitution
                0; if X(i) = Y(j)
        }
        ptr(i,j) = {
            LEFT   insertion
            DOWN  deletion
            DIAG  substitution
        }
    
```



Pawan Goyal (IIT Kharagpur)      Spelling Correction: Edit Distance      Week 2: Lecture 1      18 / 20

So, simple idea whenever so you can say these are deletion, insertion and substitution. So whichever gives you the minimum cost; you will add that pointer if all three are giving you the same cost, you might have all the three point.

(Refer Slide Time: 29:24)



So, in general suppose I start with a word X 0 to X N, so 0 means null X N is the complete word X and I have the output string Y 0 to Y M. So, if I see this matrix of size N cross M, so every possible path from 0 0 to M N is a possible alignment. So, what I am trying to find out which path gives me the minimum possible edit distance and this as per

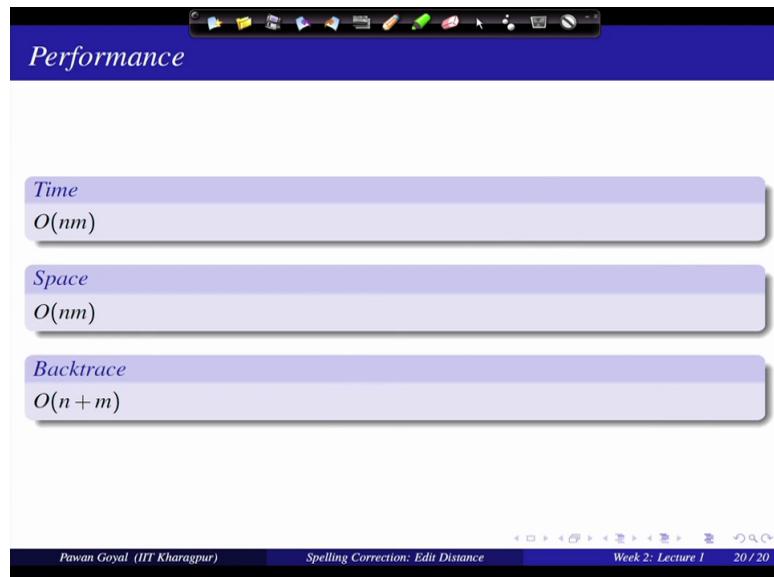
dynamic programming approach and optimal alignment should be composed of optimal sub alignments so; that means, whenever I am going from I to j, I can use always the optimal sub align alignment would be I minus 1 j and. so on, so that is how this problem algorithm is designed.

(Refer Slide Time: 30:24)

The screenshot shows a presentation slide titled "Result of Backtrace". The slide displays two strings of text, "INTE \* NTION" and "\* EXECUTION", aligned vertically. Vertical lines connect the characters in "INTE" to the corresponding characters in "\* EXECUTION". The slide has a dark blue header and footer. The footer contains the text "Pawan Goyal (IIT Kharagpur)", "Spelling Correction: Edit Distance", "Week 2: Lecture 1", and "19 / 20".

So, if you do the victories; you can find out from the intention, you go to executions and what are all the places where you did insertion, deletion and substitutions and which words you did not have to change at all, now what is the complexity of this algorithm. So, let us talk about the time complexity; how much time we took. So, you see you have to fill all the N cross M entries of the matrix. So, there are N cross M different things you have to do and for each entry, you need a constant number of operations you will find out minimum of three different distances.

(Refer Slide Time: 31:11)



So there are N cross M times some constant, so the time complexity can be simply order of N M ( $O(nm)$ ), what would the space complicity again the space complexity also the same you are using only N cross M different element that must the space that you need you are not using any other space remember for each element you are using some other elements from the same matrix. So, you not any edit distance space. So, again a space complexity is N cross M; now what about the backtrace.

So, that is I am at  $D_{n,m}$  and I want to find backtrace, the worst case can be I have to go through all the N elements in my column N elements sorry N elements in my top and M elements in my first column. So in the worst case, I might have to do N plus M different operations, so this complexity is again N plus M ( $O(n+m)$ ). So, you will see this is very very efficient in compared to the approach that favour the naive approach that you have seen earlier you start with the input strings and try out all possibility this will be become explanation very very soon.

So, the first lecture for this week and in the next lecture, we will start talking about certain variation of edit distance. Why should I even think of some variations and how do we use that for finding edit distance.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute Technology, Kharagpur**

**Lecture - 07**  
**Weighted Edit Distance, Other Variations**

Hello everyone and welcome to the second lecture of second week. So in the last lecture we talked about what is the basic edit distance concept given 2 strings as input, what is a dynamic programming algorithm that you can use find out, what is the edit distance between these 2 strings. And remember we defined certain operations, certain edit operations and we gave weights or cost for each of these operations. But all these cost for equal. If you are substituting a character by another character irrespective what are the 2 characters? The cost remains the same. Same goes for inserting a character. If the cost remains same and also for deletion of a character it remains same so in this lecture we will briefly discuss that is there any way in which we can try to discriminate between various sorts of edit operations.

Firstly is this, should we discriminate between various operation, should there we assigned the same cost or should be try to give different cost and different operations. Then we will also talk about the practical problem that given a query, Query I mean a term that might be incorrectly spelt, how do I come up with list of terms that are actual, that for actual terms that should have been used instead of the incorrect so what should be, what is a good or efficient algorithm to find out all the set of terms that are variation of edit distance of say edit distance of 1 or 2 of that query.

So starting with the concept of weighted edit distance so can we weight the edit distance cost depending on what are the characters that are involved in the operation. So firstly, is it required? So, do you think substituting a by e should have the same cost substituting a by m so what should depend on? What kind of; so for example, there are certain errors that you make very commonly and certain errors that you make very rarely. Suppose error is very common should I assigned it a higher cost or a lower cost? So think about it. What is it mean to an assigning it a higher cost? Assigning it a higher cost means that the edit distance between the 2 strings will increase if that edit operation is present. So that means, the probability of obtaining the other string as a candidate we will go down if the

operation as a high cost, but if something is some sort of edit operations are very common, some spelling mistakes are very common we would like to give them a lower cost so that I can the actual candidate have a small edit distance with the incorrect word

So, it is important to understand what kind of characters will have a lower edit distance and what kind of character will have a higher edit distance. Can be somehow give a weight. So this is the basic idea.

(Refer Slide Time: 03:52)

*Weighted Edit Distance*

*Why to add weights to the computation?*

- Some letters are more likely to be mistyped.

Pawan Goyal (IIT Kharagpur)    Weighted Edit Distance, Other variations    Week 2: Lecture 2    2 / 11

So, why do we need to add various weights to the computation? So the idea is that some letters in a language are more likely to be mistyped than others.

(Refer Slide Time: 04:10)

		Confusion Matrix for Spelling Errors																									
		sub[X, Y] = Substitution of X (incorrect) for Y (correct)																									
		Y (correct)																									
X		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	0	0	7	1	342	0	0	2	118	0	1	0	0	3	76	0	0	1	35	9	9	0	1	0	5	0	0
b	0	0	9	16	0	2	1	0	0	0	0	0	3	11	5	0	10	2	5	39	40	1	3	7	1	0	0
c	6	5	0	16	0	9	5	0	0	1	0	0	2	7	2	0	1	0	43	30	22	0	0	4	0	2	0
d	1	10	13	0	12	0	5	5	0	0	2	3	7	2	0	1	0	43	30	22	0	0	4	0	2	0	
e	388	0	3	11	0	2	2	0	89	0	0	3	0	5	93	0	0	14	12	6	15	0	1	0	18	0	
f	0	15	0	3	1	0	5	2	0	0	0	3	4	1	0	0	0	6	4	12	0	0	2	0	0	0	
g	4	1	11	11	9	2	0	0	0	1	1	3	0	0	2	1	3	5	13	21	0	0	1	0	3	0	
h	1	8	0	3	0	0	0	0	0	0	2	0	12	14	2	3	0	3	1	11	0	0	2	0	0	0	
i	103	0	0	146	0	1	0	0	0	0	6	0	0	49	0	0	0	2	1	47	0	2	1	15	0		
j	0	1	1	9	0	1	0	0	0	0	2	1	0	0	0	0	0	0	5	0	0	0	0	0	0	0	
k	1	2	8	4	1	1	2	5	0	0	0	5	0	2	0	0	0	6	0	0	0	4	0	0	0	3	
l	2	10	1	4	0	4	5	6	13	0	1	0	0	14	2	5	0	0	11	10	2	0	0	0	0	0	
m	1	9	7	8	0	2	0	6	0	0	4	0	18	0	0	6	0	9	15	13	3	2	3	0	0		
n	2	7	6	3	0	0	19	1	0	4	35	78	0	0	7	0	28	5	0	0	1	2	0	2	0		
o	91	1	1	3	116	0	0	0	25	0	2	0	0	0	0	14	0	2	4	14	39	0	0	0	18	0	
p	0	11	1	2	0	6	5	0	2	9	0	2	7	6	15	0	0	1	3	6	0	4	1	0	0	0	
q	0	0	1	0	0	27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
r	0	14	0	30	12	2	2	8	2	0	5	8	4	20	1	14	0	0	12	22	4	0	0	1	0	0	
s	11	8	27	33	35	4	0	1	0	1	0	27	0	6	1	7	0	14	15	0	0	5	3	20	1		
t	3	4	9	42	7	5	19	5	0	1	0	14	9	5	5	6	0	11	37	0	0	2	19	0	7	6	
u	20	0	0	44	0	0	64	0	0	2	43	0	0	4	0	0	0	0	2	0	8	0	0	0	0	0	
v	0	0	7	0	0	3	0	0	0	0	1	0	0	1	0	0	0	8	3	0	0	0	0	0	0		
w	2	2	1	0	1	0	2	0	0	1	0	0	0	7	0	0	6	3	1	0	0	0	0	0	0		
x	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0		
y	0	0	2	0	15	0	1	7	15	0	0	0	0	2	0	6	1	7	36	8	5	0	0	1	0	0	
z	0	0	0	7	0	0	0	0	0	7	5	0	0	0	0	2	21	3	0	0	0	0	3	0	0		

Pawan Goyal (IIT Kharagpur)

Weighted Edit Distance, Other variations

Week 2: Lecture 2 3 / 11

So, let us take some statistics. So this is a confusion matrix for spelling errors that was found in a corpus. So x is the incorrect character and y is the correct character. So what is this stability is going. How often in that corpus they found an incorrect x for a correct y. So for example, if you try to read the table, the entry for the first row, the anti-corresponding to a e, that means, how many times instead of the correct e and incorrect a bar substituted. And that number comes up to be 342. So that means in the corpus 340 times, at an instead of e a was written. Similarly, 118 times instead of i, a was written. Similarly, 76 times instead of o, a was written.

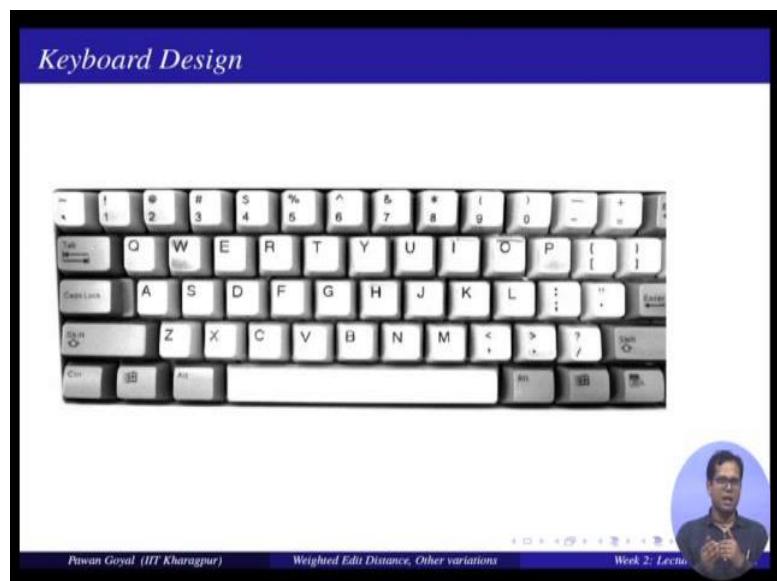
On the other hand, if you see b, it never happened that a b was written as a. So that means, people make the mistake of converting or instead of writing e writing a, this is a very common mistake on the other hand b to a is not very common. So you can look at this confusion matrix and you will find a lot of nice patterns. So one thing you will see is that the errors among the vowels are very common. So substituting i for e, e for I, a for o they are very common. So there may be because the person is may be not knowing the spelling or so is even if you know the spelling by mistake is typing the other verbal. So this is also possible.

Now, there are some other kinds of errors that you can see here that come because of the way keyboard also designed. So you will see the characters that that come very close in the keyboard, so sometimes you miss type them. So this is again a common source of a

spelling error. So for example, m and n are very close together, and they also sound very similar. So it is very likely that you mistake m for n, or n for m. So that is this kind of errors, are very common. So some errors come because some of the characters' sound very similar, so like the all the vowels or certain characters. And also some characters are very close in keyboards. So that is again another source of errors.

So, now once we know that some errors are more likely than others, can I use that this statistic to design my weighting scheme. So as I said earlier if some error is more likely, the edit cost from one character to another character should be low so that I can easily find what should be the actual candidate given in the erroneous word. So the smaller edit distance means that is more likely to be the correct candidate. That is why we will give a smaller cost to those spelling mistakes that are more likely and a larger cost or a higher cost to those spelling mistakes that are not so like, that are very rare.

(Refer Slide Time: 07:36)



So, keyboard design is again one thing that gives rise to many of the errors that we have seen in the previous slide. So again try to correlate among the characters that are coming very close in the keyboard. So we will see that many a times people make mistakes in typing one character for another.

(Refer Slide Time: 07:59)

*Weighted Minimum Edit Distance*

**Initialization:**

$$D(0, 0) = 0$$
$$D(i, 0) = D(i-1, 0) + \text{del}[x(i)]; \quad 1 < i \leq N$$
$$D(0, j) = D(0, j-1) + \text{ins}[y(j)]; \quad 1 < j \leq M$$

**Recurrence Relation:**

$$D(i, j) = \min \begin{cases} D(i-1, j) + \text{del}[x(i)] \\ D(i, j-1) + \text{ins}[y(j)] \\ D(i-1, j-1) + \text{sub}[x(i), y(j)] \end{cases}$$

**Termination:**

$$D(N, M) \text{ is distance}$$

Navigation icons: back, forward, search, etc.

Pawan Goyal (IIT Kharagpur)    Weighted Edit Distance, Other variations    Week 2: Lecture 2    5 / 11

So, now suppose you have some data and by analyzing the data you can find out what kind of errors are common, what kind of errors are rare. And accordingly you can design your weights for different edit operations. So now, once you have each weight that depends on the actual characters you can modify your initial algorithm. So how can you modify your algorithm? So for example, here so I will, so if you if you try to compare this with an algorithm that we saw in the previous slide, small reference here is that we should have giving a uniform cost for deletion insertion substitution.

What we are doing here so deletion is now conditioned on the actual character that is been deleted. So you see  $D(i, 0)$  ( $D(i, 0)$ ) is  $D(i-1, 0)$  plus a cost for deletion of  $x_i$ . So what is the cost for deleting that particular character? Same goes for insertion. What is the cost for inserting that particular character? Even in substitution you might have cost for so that separate cost for deletion insertion and substitution of one character by another. And everything else remains the same the only thing that changes is that instead of using equivalent cost for each operation you are now giving cost that are dependent on the actual characters that are inserted or substituted or deleted. So that is my weighted edit distance.

(Refer Slide Time: 09:39)

How to modify the algorithm with transpose?

Transpose

- $\text{transpose}(x, y) = (y, x)$
- Also known as metathesis

Pawan Goyal (IIT Kharagpur)      Weighted Edit Distance, Other variations      Week 2: Lecture 2      6 / 11

So, now so what to do you do with, so how do you modify your algorithm for taking care of transpose for instance. So that is the algorithm were we had 3 operations. We had insertion, substitution and deletion. They were 3 operations that we why we are using in that previous operation. So there is another very common edit distance operation that is used and this is called transpose. So if I am transposing toward 2 characters. And you will receive this is a common error that sometimes you make.

(Refer Slide Time: 10:19)

$\underline{al} \leftrightarrow \underline{la}$

transpose (metathesis)

$\boxed{x_i y_j \rightarrow y_j x_i} \rightarrow 1 \text{ for transposition}$

$D(i,j) = \min \left\{ \begin{array}{l} D(i-1, j-1) + 1 \\ \quad \text{(transpose)} \\ \quad \text{if } X[i-1] = Y[j-1] \\ \quad \text{or } X[i] = Y[j-1] \\ \quad \text{insertion} \\ \quad \text{deletion} \\ \quad \text{substitution} \end{array} \right\}$

So, for example, so I am wanting to type al, and instead I typed it as la. So this is called

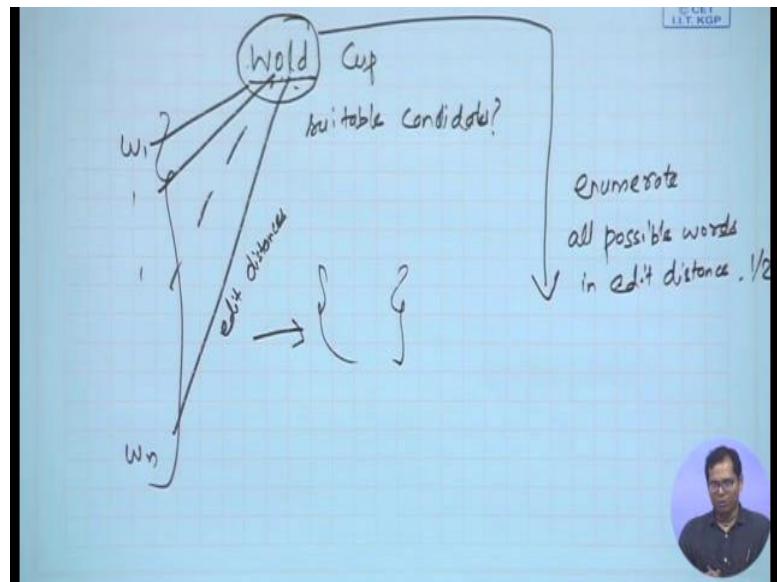
transpose. So in general if I write  $xy$  as  $yx$ . So this is called transpose another name for this is metathesis. So now, how can I modify my earlier algorithm, to take care of this case? So till now we have 3 operations insertion deletion substitution. In insertion I see the previous insertion deletion I see only till the previous word, but what do I do in the case of transposition. So in the transposition I will have to so because it is we have go to characters before so I need to go to  $D$  in my cos matrix  $D[i-2, j-2]$ ). And suppose I give a cost of one for transposition.

So, I will write  $D(i,j)$  as so there are other cost that we had defined earlier, but for transposition I will say  $D[i-2, j-2] + 1$  and what will be the condition - only if the previous word in  $x$ , so I had defined my strings as  $x$  and  $y$ . Here so this is my string as  $x$  this is my string as  $y$ . So what will happen here? This is up to  $i$  so this  $x[i-1]$  will be same as the  $j$ th character in  $y$ . So if  $x[i-1] = y[j]$  and  $x[i] \neq y[j]$  then  $D[i, j] = D[i-1, j-1] + 1$ . So if I find that so  $x[i] = y[j]$  then  $D[i, j] = D[i-1, j-1]$ . So if I find that the  $i$ th character of  $x$  is same as the  $j$ th character of  $y$ , and the  $i$ th character of the  $x$  is same as the  $j$ th character of  $y$ , then I will say that this cost is  $D[i, j] = D[i-1, j-1] + 1$ .

So, again I will do a minimum of this. This if this happens this is my transpose and then they I have other conditions for insertion deletion and substitution. And that is how I can modify my algorithm by initial algorithm to also include transposition. So with all these operation this is the only other operation that is also commonly used in computing edit distance. So now, so we talk about edit distance and also about doing including another edit operation like transpose.

So as I said earlier so we will also discuss briefly, so what is the practical scenario. So the practical scenario here is you are given an input word. So remember in the introduction slide we 2 are talking about this error.

(Refer Slide Time: 14:17)



Say wold cup and by mistake I have written wold instead of writing world. So now, the practical problem here is once I know that this is not the correct word in my vocabulary, how do I find out what are the possible candidates that might come in place of w o l d. So that is what are the suitable candidates. Now as per our definition so the suitable candidates are those that are having a small edit distance from the actual error word. So intern my problem is how go I find out words that are within some a small error some small edit distance of word.

So, now how do I solve this problem; given a word like wold find out all the words that are within the additions of say 1 or 2. Now one simple thing you might say that I will list down all the words by vocabulary somewhere, so I have all the words in my vocabulary starting from w1 to wn and I will find out what is edit distance of w 1 with world, w 2 with world, and up to wn. I will find out all the edit distances and then I will among those I will choose all those that are coming up to the very small edit distance some top entries that are coming out to be within a small edit distance. So I will find out this type of entries. This is my candidate.

But you can immediately see that this may not be a very efficient solution because my vocabulary size can be quite huge and I have to find out edit distance from this to each of the words in my vocabulary. So that will be really time consuming, so instead can I so can I do something more efficient. So one simple thing is I should try to search in an

efficient data structure for searching all the words in my dictionary.

So in case you do not know so one of the very efficient edit structures that is used for searching the strings is called the trie structure. So using the trie structure you can do search efficiently in the linear time, so linear in the length of the input string.

(Refer Slide Time: 16:53)

*How to find dictionary entries with smallest edit distance?*

*Naïve Method*  
Compute edit ditance from the query term to each dictionary term – an exhaustive search

*Can be made efficient if we do it over a trie structure*

Pawan Goyal (IIT Kharagpur)      Weighted Edit Distance, Other variations      Week 2: Lecture 2      7 / 11

So, I am giving you one example here. So I can use the trie structure. So idea is that so you take any word you can start from the initial root node of this trie structure and keep on following the arcs and in the time linear to the length of your word you will reach to a node and find out if the word is available in the vocabulary or not. So by that you will able to search all the words in the vocabulary. And this trie structure can be used efficiently for computing the distance of any new word with a word in this trie. So, this is one possibility.

But this again it requires you to do a lot of computations with respect to all the words in your vocabulary. What can be other possibility? So other possibility if your thing would be so instead of trying out to find out the edit distance of this error word to all the words in the dictionary, can I start from my error word and try to enumerate all possible words within some edit distance, say 1 or 2 I am trying to enumerate all the possible words edit distance 1 or 2. So I will find out the word like world and everything else and then I can have a further check to see if these words there in my dictionary and all, and there I can use maybe a trie structure, that whether these words are available in my dictionary or not.

So, this can of the possibility. Now what is, do you see do think this will be efficient this approach. So let us see some numbers.

(Refer Slide Time: 18:57)

*How to find dictionary entries with smallest edit distance?*

- Generate all possible terms with an edit distance  $\leq 2$  (deletion + transpose + substitution + insertion) from the query term and search them in the dictionary.
- For a word of length 9, alphabet of size 36, this will lead to 114,324 terms to search for
- For Chinese alphabet size is 70,000 (Unicode Han Characters)

Pawan Goyal (IIT Kharagpur)    Weighted Edit Distance, Other variations    Week 2: Lecture 2

So, yes so I can possible generate all possible terms is starting from the error word that are within edit distance of less than equal to 2. And there I can take care of all the deletion substitution addition and transposition. I can try to take care of all the possible edit operations. So starting where wold I generate all the possibilities with the edit distance of 2.

Now, so if you just try to do it is a simple math, try to compute how many different possibilities you will have to explore. So let us see. So suppose if you take a word of length 9 that is 9 characters and suppose in you are taking a language like English and alphabet is 36 and so that 26 plus other characters. And if you just do, that so this will give you roughly 114,324 different possibilities to explore staring from the word of length 9 in which of the 4 you think will give you a lot of possibilities. Deletion will probably not give you many possibilities, but substitution will give you lot of possibilities and insertion should also give you lot of possibilities.

Substitution you can substitute each character by and you have 36 characters. So that gives you a huge number of possibilities. So this is again huge number. So it is starting from a word of length 9 you are going to explore 114,324 different possibilities, but English is still ok. So you have 36 characters. So think about a language like Chinese

where the alphabets size is 70,000. So you cannot even think of applying an approach like that.

So, this is an approach is slightly better than then before, but it is may not be very efficient again, depending on the alphabet size. So what can be the other possibilities? So other possible algorithm is that the idea that all this edit distances can be attained if you try to do a symmetric delete from the current word and the possible candidate words, so all the edit distance of operation 2 can be found out just by doing deletes in both the words.

(Refer Slide Time: 21:26)

The slide has a blue header bar with the text "How to find dictionary entries with smallest edit distance?". Below the header is a purple box containing the title "Symmetric Delete Spelling Correction" and two bullet points:

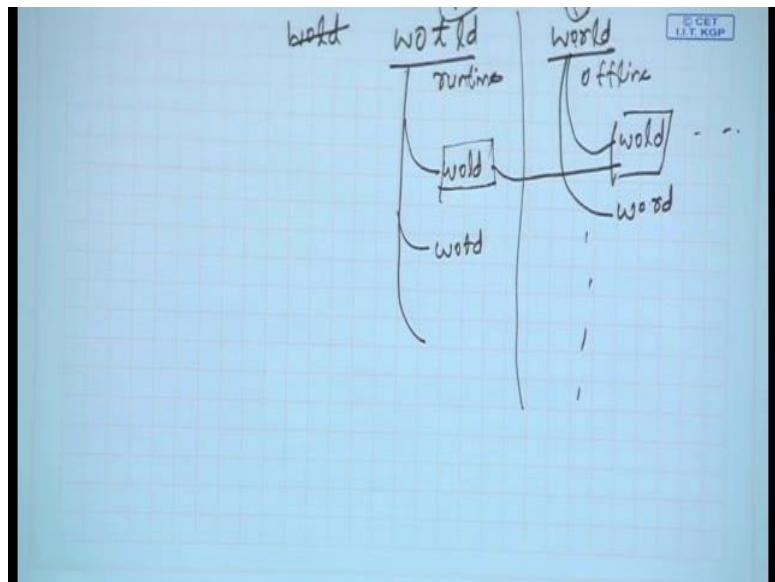
- Generate terms with an edit distance  $\leq 2$  (deletes) from each dictionary term (offline)
- Generate terms with an edit distance  $\leq 2$  (deletes) from the input terms and search in dictionary

At the bottom of the slide, there is a footer bar with the following information:  
Pawan Goyal (IIT Kharagpur)    Weighted Edit Distance, Other variations    Week 2: Lecture 2    9 / 11

So, what is the idea? So this is called the symmetric delete a spelling correction. So idea is that from your query word and so from your query word you will try to find out all the words that come up within delete of 2. From the query word and from your dictionary for each of the word you will try to find out all possibility that come within and edit distance 2 as per only the deletes. And that this will cover all the possible variations of edit distance of 2.

So, you can we can take a simple example. So let us take the same example of wold or so even or we can try to take a different example.

(Refer Slide Time: 22:05)



Say wotld instead of world that is the correct word and this is the error word. So what we will do is symmetric delete operation. So this is this I will do offline. It is an offline a starting from word I will store all the possibilities within edit distance of 1 and 2. So I will store with world that there is word wotld that comes up word and so on and from these again you will have this is delete 1 and you will also have again either candidates for delete 2. At run time when you get this query like wotld that is incorrect, you will do the same thing. So this is at run time and you will again generate all the possibilities with delete of 1 and 2.

So, we are doing here one only to show the possible. So you will find wold, wotd and so on. And now you will try to match if one of this is available in this dictionary that is built by this symmetric delete operation. And you will find wold and wotd is a match. And once this is a match you will go back to the original word, and then confirm that this word and this word have an edit distance of less than or equal to 2. And that is how you can find out all the possible words within edit distance of 2, without I am doing a lot at run time. Now can you tell why; because you only doing deletes at run time and deletes a not mind.

So, if you do one delete if you heard of 9 characters you have only 9 possible deletes. So this is not very inefficient at run time and this all is done at offline. And later you will have a further check to see that these 2 words have additions of 2. If your 2 or 3 sort. So

yeah if you take number of deletes of 2 for a word of length 9 it will be at most 45. So this is not a very big number. So this is one approach that you can use for finding out all the possible dictionary terms within some edit distance efficiently at run time.

(Refer Slide Time: 24:55)

*Non-word spelling errors*

*Non-word spelling error detection*

- Any word not in a dictionary is an error
- The larger the dictionary the better

*Non-word spelling error correction*

- Generate candidates: real words that are similar to the error word
- Choose the best one:
  - > Shortest weighted edit distance
  - > Highest noisy channel probability

Pawan Goyal (IIT Kharagpur)      Weighted Edit Distance, Other variations      Week 2: Lecture 2      11 / 11

So, now so when we talk about edit distance and some variations. So we will not focus on the task of spelling correction with which we had started this module on edit distance and all. So when we talk about spelling errors there are 2 kind of errors that you might have to deal with. So one that are easy to steep to handle errors, so I will say slightly easier they are called non word errors because they are easy to detect. So the erroneous word is not in my vocabulary.

So in this case so erroneous word is b e h a f. That is not in my vocabulary I can deduct decision erroneous word and my task is to correct it, to the actual word b e h a l f, behalf and this is also called non word error. What is slightly more difficult is something called a real word error. When the word that is in error is also in my dictionary, but somehow the user has missed spelt to a word that is in my dictionary so here some examples. So like typographical error so instead of writing there somebody has write 3, now 3 in my vocabulary so it is not easy to that this is in error, now correcting this is called a real word error correction.

Similarly, they can be something for because of the homophones, because of say piece and peace the 2 different variations of pieces you might misspell very easily. And too and

two is very common error. So they are also called real word errors. Now to handle both of the non-word error and real word error, you need to use various probabilistic models. And this is what we will be focusing. So in non-word spelling errors so the non-word means the word that is not in my edit dictionary, so any word that is not in the dictionary is called an error. And so for handling this problem it is better that you have a good very good lexicon tells you all the possible word in your lexicon. And this lexicon may also be dependent on the particular corpus that you are processing so, for you if you are processing scientific corpus, you might have different corpus, then if you are processing use corpus. So you should know what your words in the in the dictionary and when you encounter word that is not in your dictionary you will treated as an errors and try to correct it.

So in general what you will do in non-word spelling correction? So starting from the error word you will try to find out candidates that are similar to the error word, and how will you define the similarity with respect to the error word. So you will say the words that are within form a small at a distance. That can be one possible criteria and if you are using noisy channel model, that will be the topic for the next module, so you will see the one that is coming out to be as per the highest noise and probability. So this will be your candidate words. So you will choose some of the candidate words like that.

(Refer Slide Time: 28:12)

*Real word spelling errors*

```

graph TD
    A[For each word w, generate candidate set] --> B[Choosing best candidate]
    B --> C[Noisy Channel]
  
```

*For each word  $w$ , generate candidate set*

- Find candidate words with similar pronunciations
- Find candidate words with similar spelling
- Include  $w$  in candidate set

*Choosing best candidate*

- Noisy Channel

Pawan Goyal (IIT Kharagpur)      Weighted Edit Distance, Other variations      Week 2: Lecture 2      12 / 11

On the other hand, if you are dealing with the real word spelling error, you will try to

find out so now you have the; you did not know in the sentence which of the word is actually erroneous. So what you will do for each word you will find out other words that are similar in edit distance or some other criteria, but in the candidate you will also include your actual word. So two is there so we will include too also, with two and you say that the actual sentence might contain any of these 2 words. And then finally, you will try to maximize the probability of the sentence using some noisy channel model. So this is what we will be discussing in the next lecture, where we will talk about noisy channel model in detail and how it can be used for doing the task of spelling correction.

Thank you.

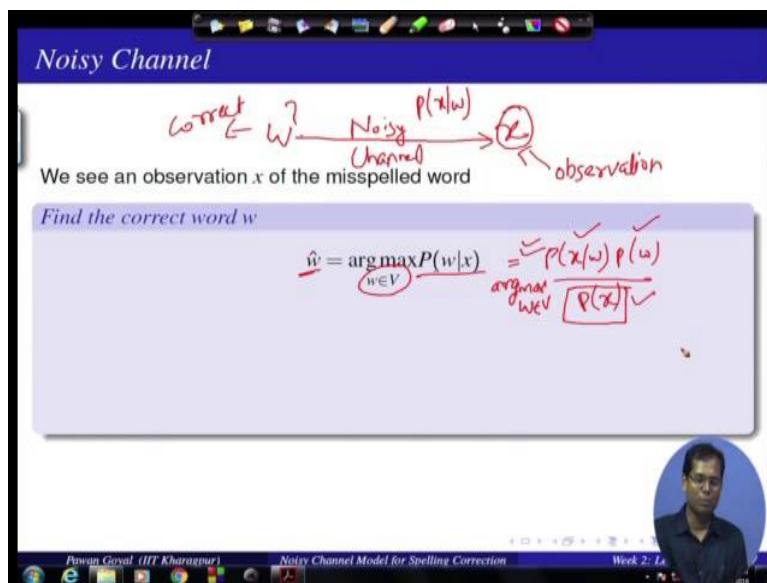
**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute Technology, Kharagpur**

**Lecture - 08**  
**Noisy Channel Model for Spelling Correction**

Welcome back for the 3rd lecture of this week. So, in the last lecture, we talked about what are the various variations of edit distance that you might be using in general. So, today we will now talk about very extended algorithm for doing spelling correction.

So, in general when I talk about spelling correction, I might be talking about isolated word reduction or correction that is a word without the context, even there it might be a word that is in my vocabulary or not in my vocabulary. So, we will see examples for that or I might talk about doing corrections in the context, there again a word might be in the vocabulary or out of the vocabulary and we will see how the simple approach of noisy channel model can be applied for doing a spelling correction.

(Refer Slide Time: 01:19)



Now, what is my noisy channel model? So, if you think of the model name, what is happening? We are having a observation  $x$  that is a word, that is misspelled and you wanted to correct it that is your observation, but what is the idea of this model? So, whenever have a misspelling you wanted to write a correct word and you went through a channel that can be your keyboard or something else and that is how your word got misspelled. So, that is what you are assuming

you have a correct word  $w$ , this is your correct word and you have an incorrect word  $x$  that is your observation, yes, now you wanted to type this correct word  $w$ , but you went through a channel and this we are saying is a noisy channel and you are making some mistake while going through this channel and the word that you are observing is  $x$ , but not  $w$ . Now your observation is  $x$ , now given this observation  $x$  how do I find out what is my mostly slightly candidate for  $w$ ? What is my  $w$ ?

In general how can I try to solve this problem? I am seeing a word  $x$  that I know it is misspelt, now how do I know it is misspelt? A crude way of doing that is I maintain a dictionary that is my set of all the words in my vocabulary and if the word  $x$  does not match with any of these words; I say this might be in misspelled. Now once I know  $x$  is a misspelled word; that means, I have to correct it, now what are we have to first start with what are all the possible candidates for correction. So, suppose possible there could be  $\hat{w}$  find out all the words within the added distance of something now among those which one is the most slightly candidate. So, so in terms of noisy channel model what is the  $w$  from which  $x$  has been written. So, misspelt and we wrote  $x$ . So, in terms of noisy channel model I want to find out  $\hat{w}$  that gives me the maximum probability  $P(w|x)$  among all the possible words in my vocabulary or it can be in my set of candidates what is the  $\hat{w}$  that gives me the maximum value for  $P(w|x)$   $\hat{w} = \arg \max P(w|x)$  where  $w \in \mathcal{V}$ .

Now, how do I compute this probability  $P(w|x)$ ? You see as for the noisy channel model I first started with the word  $w$  and then I went through the channel and I ended up with the word  $x$ . So, I can only find out using my channel what is the probability of  $x$  given  $w$ , I will have to somehow used to find out probability of  $w$  given  $x$  yes. So, what is a particular theorem in probability that you can use for is. So, remember Bayes theorem. So, I want to find out probably  $w$  given  $x$ , but I already can find out from my channel probability of  $x$  given  $w$ . So, how do I write  $P(w|x)$  in this term? So, I say this will be  $P(x|w)P(w)/P(x)$  yes  $(P(x|w)P(w))/P(x)$ . So, I can replace  $I$  by  $\arg \max$  of this.

Now what is your variable here? The variable is  $w$  you can have multiple different words, but your  $x$  remains the same for each individual word. So, what, in fact, you are trying to do? For each word you are completing this value and then you are seeing that is the maximum value which word gives you the maximum value. Now among the 3 different probabilities that you are using for this formula which one you might remove for this computation? Is it the probability  $P(x)$ , remains the constant across all the words. So, it will not matter to my decision

of which one is having the highest probability because this was the simply depend on this particular multiplication. So, I can as well remove  $P(x)$  and keep my argument and that is how we actually use.

(Refer Slide Time: 06:03)

We see an observation  $x$  of the misspelled word

*Find the correct word  $w$*

$$\begin{aligned}\hat{w} &= \arg \max_{w \in V} P(w|x) \\ &= \arg \max_{w \in V} \frac{P(x|w)P(w)}{P(x)} \\ &= \arg \max_{w \in V} P(x|w)P(w)\end{aligned}$$

Pawan Goyal (IIT Kharagpur)      Noisy Channel Model for Spelling Correction      Week 2: Lecture 3      2 / 17

I write it like that and I remove the probability  $P(x)$  and that is how that is. So, how I will find out which one is the correct word  $w$  for  $x$  which give me the maximum probability for probability  $x$  given  $w$  times probability  $w$  ( $\text{argmax } P(x|w)P(w) \text{ where } w \in v$ ). So, now, we will see how do we compute these probabilities individually what are the various which we can compute this probabilities.

(Refer Slide Time: 06:30)

Non-word spelling error: across

Words with similar spelling  
Small edit distance to error

Words with similar pronunciation  
Small edit distance of pronunciation to error

Damerau-Levenshtein edit distance  
Minimum edit distance, where edits are:  
Insertion, Deletion, Substitution,  
Transposition of two adjacent letters

Pawan Goyal (IIT Kharagpur)      Noisy Channel Model for Spelling Correction      Week 2: Lecture 3      3 / 17

So, we are taking about non word spelling errors. So what do I mean by non word spelling errors? A word that is not in the dictionary or my vocabulary and I know this might be misspelled word, suppose I have taken the word here actress. So, actress is not in the dictionary it can be actress across or whatever, but we do not know. So, this is the observation that we made, now using noisy channel model I want to find out what should have been the correct word. So, now, the first thing I will needed to do I will need to find out what are the candidate words that might be correct.

How do I find the candidate words I will try to find out words that are having similar spellings that are having small edit distances or I might try to use the words that are having similar pronunciation? So, both of these are possible. So, then I make some sort of candidate set. So, suppose I have ten different words that might have given to actress. So, these are my various candidates for W and I need to use the noisy channel model to find out which one is the most slightly candidate.

And suppose I am using might d 1 at a distance this is small variation of my Levenshtein distance remember Levenshtein distance what were the operations, we at insertion deletion and substitution, in this particular edit distance we also have the transposition. So, insertion, deletion, substitution and transposition of 2 adjacent correct words. So, we discussed about how to how to modify our dynamic problem algorithm to take into count of this transposition

in the last lecture. So, now, so given 2 strings, I can always find out what is the word. So, given a string I can find out what are the other words that come within additions of 1 or 2.

(Refer Slide Time: 08:36)

Error	Candidate Correction	Correct Letter	Error Letter	Type
acress	actress	t	-	deletion
acress	cress	-	a	insertion
acress	caress	ca	ac	transposition
acress	access	c	r	substitution
acress	across	o	e	substitution
acress	acres	-	s	insertion
acress	acres	-	s	insertion

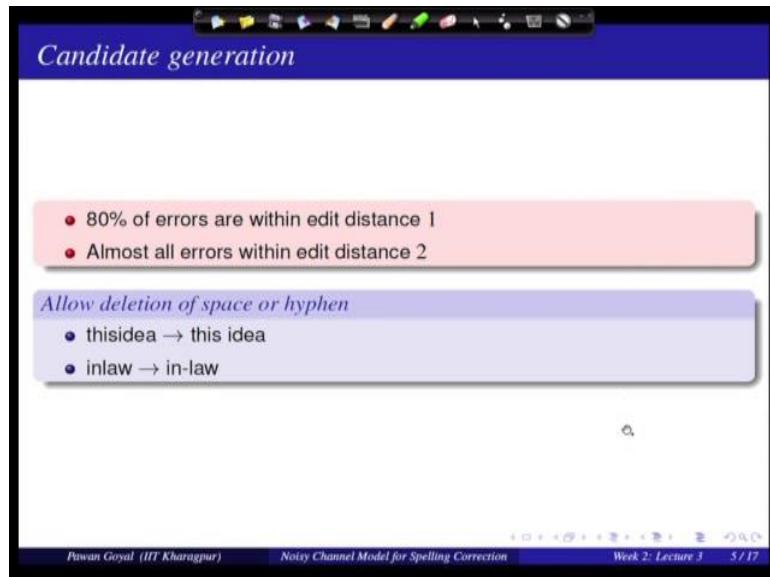
Suppose I start with acress that is my observation; incorrect word, I try to find out that are the also candidates are come with additions of 1.

Here is I have, here is the word acress, now I find the words like actress, cress, caress, access, across, acres, and acres occurs twice in this list, why and we will see that. Let us try to understand first candidate word actress. So, what does we have written here? So, we have written. So, this is my correct word this is my incorrect word. So, what change did I make for going from correct word to incorrect word? So, I went I had a letter t in my correct word, but I replace it with null. So, this is a deletion, I delete it; t from my correct word. So, that is why this is deletion.

Similarly, how would I go from cress to actress, I have to have insert a. So, this is the again insertion, yes, similarly, how I go from caress to acress I am doing a transposition. So, ca change to ac transposition - access to acress; c got change into r across to acress o got change to e now. Why there were 2 acres? you see acress I can go by doing 1 insertion, but insertion can happen at 2 points it can happen either here or it can happen here and these 2 will have different probabilities. remember with the probabilities of substitution will depend on what is the previous word that what is the previous character in this word. So, they will different probabilities, that is why they are 2 different insertions.

Now, that means, whenever I have a misspelled word I have to first generate all the possible candidates. So, there we try to use some sort of a statistics how many different candidates should I be using. So, the general statistics is that 80 percent of the errors are within distance of one and nearly all the errors are within the additions of 2 so; that means, I can mostly try to use the candidates within additions of 1 or 2 may be, one will do for most of the cases.

(Refer Slide Time: 11:24)



That is what we did in the previous example starting from across we found out all the candidates with in additions of one and while generating the candidates I might also allow for deletion of space or hyphen. So, if I a word like this idea without a space, I might allow this idea with this space and similarly for hyphens and in law to in-law with a hyphen.

(Refer Slide Time: 11:45)

Computing error probability: confusion matrix

- del[x,y]: count (xy typed as x)
- ins[x,y]: count (x typed as xy)
- sub[x,y]: count (x typed as y)
- trans[x,y]: count(xy typed as yx)

Real errors:  $x-y$  got typed as  $x$ .  
Correct:  $x-y$  got typed as  $y$ .

Pawan Goyal (IIT Kharagpur) Noisy Channel Model for Spelling Correction Week 2: 1a

Now when we are trying to compute the probabilities, we had 2 different terms probability of  $x$  given  $w$  and probability of  $w$ . So, you will first see how do we compute probability of  $x$  given  $w$ . Now suppose we are taking the simple case where there are only one this only one edit operation from the input string or the correct word to the incorrect word.

What do I mean by probability of  $x$  given  $w$ ? I would mean what is the probability of that edit operation being done. So, remember we had four operations, insertion, deletion, substitution and transposition. So, we need to define probabilities for each of these. So, here that is what we are defining we are defining deletion over  $x$   $y$  what do I mean by that? Whenever in the input string a input word I had the sequence  $x$   $y$  this is my correct and it got typed as  $x$  this is my incorrect. So,  $x$   $y$  got typed as  $x$ . So, this is I have deleted  $y$ . So, this will depend on the previous character  $x$  in the correction.

Similarly, what about insertion I had  $x$  in the correct and I inserted  $y$  in the incorrect. So, this is the case of insertion substitution  $x$  got typed as  $y$ . So, I had  $x$  in my correction I got typed as  $y$  in my incorrect decision. What is transposition? Again  $x$   $y$  here incorrect got typed as  $y$   $x$  in my incorrect. So, these are the four different possibilities if I am having only one edit operation between the 2 strings. So, now, the question is how do I find out probability of a particular deletion or insertion operation. So, let us take one of these and see how we will actually find out find out these probabilities.

Let us take the simple case of deletion. So, how do we find the probability that in a string, a set of characters x y, a character bigram, I can also say it is a hecta bigram got converted to only x.

How I will feel find out this probability. So, we will have to see in general in corpus how likely is this error? So, now, what kind of data do I need to count to find this probability? So, I need some sort of data were you just have written some texts and they made some mistakes not knowingly. So, they made some mistakes that if general people do and somebody has then seen this corpus and tried to correct it. So, suppose I have a corpus that is corpus is a real corpus and that contains various errors and then I also have a corrected corpus.

Now, how will I use these 2 different corpora to find out the probability of deletion x y, I will say how many times do I seen my corrected corpus a word containing the bigram x y for which the corresponding word in my actual real corpus had only x; that means, actually the word should have been having x y, but it was remains x. So, how will compute the probability? I will see in my real corpus how many times I have this corrected bigram x y and out of those how many times y was deleted in my real corpus and that will give me the probability that how after x what is the probability that y is deleted. So, whenever how many times x y occurred together out those how many cases y was deleted. And similarly I will find out find it out for all cases of insertion, substitution and transposition.

Let us see the actual matrices.

(Refer Slide Time: 16:23)

Computing error probability: confusion matrix

- del[x,y]: count (xy typed as x)
- ins[x,y]: count (x typed as xy)
- sub[x,y]: count (x typed as y)
- trans[x,y]: count(xy typed as yx)

Insertion and deletion are conditioned on previous character

Piwan Goyal (IIT Kharagpur) Noisy Channel Model for Spelling Correction Week 2: Lecture 3 6 / 17

(Refer Slide Time: 16:27)

$$P(x|w) = \begin{cases} \frac{\text{del}[w_{i-1}, w_i]}{\text{count}[w_{i-1} w_i]}, & \text{if deletion} \\ \frac{\text{ins}[w_{i-1}, x_i]}{\text{count}[w_{i-1}]}, & \text{if insertion} \\ \frac{\text{sub}[x_i, w_i]}{\text{count}[w_i]}, & \text{if substitution} \\ \frac{\text{trans}[w_i, w_{i+1}]}{\text{count}[w_i w_{i+1}]}, & \text{if transposition} \end{cases}$$

So, you are seeing here insertion, deletions or conditioned of the previous character. So, that is how I find out the probability of  $x$  given  $w$   $x$  is my incorrect word and  $w$  is my possible candidate for correction. So, remember how we found out deletion, we said how many times this correct a bigram  $w i$  minus 1 and  $w i$  occur in my corrected corpus and among those how many times  $w i$  was deleted after  $W i$  minus 1. Similarly here insertion how many times the word  $w i$  minus one occurs in my corpus and among those how many times I inserted  $x i$  in my real corpus were people made mistake how many times they actually inserted  $x i$  after  $w I$ ,  $w i$  minus q.

Similar substitution how many times  $w i$  occurs in my corpus, among those how many times it was substituted with  $x i$ . Same thing you will do for transposition how many times these 2 characters occurred together among those what fraction of times they were transposed. That means, we need 2 corpus - one we have made mistakes another the same corpus that has been corrected and then you can compute all these probabilities.

(Refer Slide Time: 17:58)

Candidate Correction	Correct Letter	Error Letter	x w	P(x word)
actress	t	-	c ct	.000117
cress	-	a	a #	.00000144
caress	ca	ac	ac ca	.00000164
access	c	r	r c	.000000209
across	o	e	e o	.0000093
acres	-	s	es e	.0000321
acres	-	s	ss s	.0000342

Now, if we go back to the previous example of finding various candidates for actress. So, we had 7 different candidates here.

Now, we already saw what is the corrected error word and we are using the model, we saw in the previous slide to find out probability of x given word this one. So, now, first we are writing what is my x given w? So, whenever this first my deletion from actress t got delete and we got actress so, how did you find the probability of deletion? We said what is the probability that ct, among all the times that ct occurs what is the probability that t gets deleted. So, out of ct, I see only c in my corpus and this probability, I find my corpus to be 0.000117 similarly insertion. That means, a got inserted in the beginning of the string transposition ac in place of ca and so on in so, here we have you see there are 2 different ways you can do, insertion after e or after s, that is why you have 2 different probability that is also.

Again from the corpus, you will find all these probabilities values; probability of x given my word for different words what is probability x given now what is the other probability of the compute probability of w itself. So, what is the intuition as such? You will try to give try to favor a word that is more likely to occur in the corpus than a word that is not so likely in the corpus.

(Refer Slide Time: 19:59)

Candidate Correction	Correct Letter	Error Letter	x w	P(x word)	P(word)	$10^9 \cdot P(x w)P(w)$
actress	t	-	c ct	.000117	.0000231	2.7
cress	-	a	a #	.00000144	.000000544	.00078
caress	ca	ac	ac ca	.00000164	.00000170	.0028
access	c	r	r c	.000000209	.0000916	.019
across	o	e	e o	.0000093	.000299	2.8
acres	-	s	es e	.0000321	.0000318	1.0
acres	-	s	ss s	.0000342	.0000318	1.0

Pawan Goyal (IIT Kharagpur) Noisy Channel Model for Spelling Correction Week 2: 1a

Suppose I use a corpus and I also find out the probability of the word. So, now, I have found out probability of  $x$  given word and the probability of the word both of this. So, remember my noisy channel model I have to multiply these probabilities and find out which of the candidates gives me the highest probability value.

Suppose I do the same. So, in this slide you will see. So, I am multiplying these 2 probabilities and we are putting 10 to the power 9 just so that we are able to see these numbers they were be a 0.000 to the point that we will not able to compare. So, the first probability is 2.7 times 10 to the power 9 minus 9 sorry. So, what are the 2 candidates that have the highest probability? You see here, I have actress and I have across. So, these are 2 words that I having the highest probability and in general you might right across has the correct candidate.

Now remember we were saying, we can do it without with the context or with the context. So, without using the context this is the best we can do and suppose they come up to be really close 2.7; 2.71 without using the context you cannot find out which is the better candidate than the other, but suppose you are allowed to use the contexts. So, can you do better? So, let us see for the same spelling correction if I have the context what will I do.

(Refer Slide Time: 21:30)

Using a bigram language model

- "... versatile across whose ..."
- Counts from the Corpus of Contemporary American English with add-1 smoothing
- $P(\text{actress}|\text{versatile}) = 0.000021$ ,  $P(\text{across}|\text{versatile}) = 0.000021$
- $P(\text{whose}|\text{actress}) = 0.0010$ ,  $P(\text{whose}|\text{across}) = 0.000006$
- $P(\text{"versatile actress whose"}) = 0.000021 * 0.0010 = 210 \times 10^{-10}$
- $P(\text{"versatile across whose"}) = 0.000021 * 0.000006 = 1 \times 10^{-10}$

Pawan Goyal (IIT Kharagpur) Noisy Channel Model for Spelling Correction Week 2: Lecture 3 10 / 17

Suppose my sentence says versatile across whose and I have to correct it and remember what were 2 candidates? The 2 candidates are actress and across. So, the 2 sentences are versatile across actress, whose versatile across who is now which one is more likely.

Now, how do I find out? How do I use the fact that probably versatile actress or actress whose is more common than versatile across and across whose so. So for that what we can do we take some corpus. So, this is some corpus of contemporary American English and we use some smoothing. So, do not worry about what I mean by smoothing. We will cover that in detail in language modeling. So, suppose I do some smoothing, so idea is that if we try to find out the probabilities of a word coming after another word that is what we are doing and from there I find out these probabilities that the word actress occurs after versatile is 0.000021 and the probability that the word across occurs after versatile is the same and that makes sense also, versatile across many different fields and versatile actress they might have the same sort of probability of occurrence in the corpus.

Now, what about the other bigram? What about the other 2 word combination that is actress whose and across whose? So, we see in the corpus we find actress whose has a much higher probability than across whose and if you have this information we can use that to refine the probabilities for that. If suppose I simply multiply the probabilities with my earlier corpus. So, what will I get? So, I will say probability of versatile actress whose is the initial the 2 probabilities 210 times 10 to the minus 10 and probability of versatile across whose will be 1

times 10 to the minus 10. So, this gives me that versatile actress whose is much more lightly than versatile actress, whose as per my corpus if I am using the context and this can further help me to find out what is the correct word that should have been there in place of actress.

(Refer Slide Time: 24:01)

The slide has a blue header bar with the title "Real-word spelling errors". Below the title are two bullet points:

- The study was conducted mainly **be** John Black
- The design **an** construction of the system ...

A pink horizontal bar contains the text "25-40% of spelling errors are real words". At the bottom of the slide, there is footer information: "Piwan Goyal (IIT Kharagpur)", "Noisy Channel Model for Spelling Correction", "Week 2: Lecture 3", and "11 / 17".

Now the next concept is the real word spelling errors. So, what do I mean by that? There might be errors where the misspelled word is actually in my dictionary. So, like you see the sentence here, the study was conducted mainly be John Black and we know the correct word should have been by, but the user ended up typing be. Now just think for a moment can I solve this problem without using the context. So, without using the context, how can I even say that it should be by and not be I might use the individual probability of a word occurrence, but that is not a good method to say that, so in that case, I might find errors everywhere in my corpus.

So, I should be try to using my context in somewhere. Same in the next sentence the design in construction of the system and I know this should be and, but the user ended up typing an now these are real words in the vocabulary and now I want to correct the spellings one thing is sure I should be using the context now how do I actually use the context for trying to correct these errors and this is again some statistics that 25 to 40 percent of the spelling errors that this here are real words.

(Refer Slide Time: 25:28)

The slide has a blue header bar with the title "Noisy channel for real-word spell correction". Below the header, there is a purple box containing the following text:  
Given a sentence  $X = w_1, w_2, w_3 \dots, w_n$   
List of candidates:

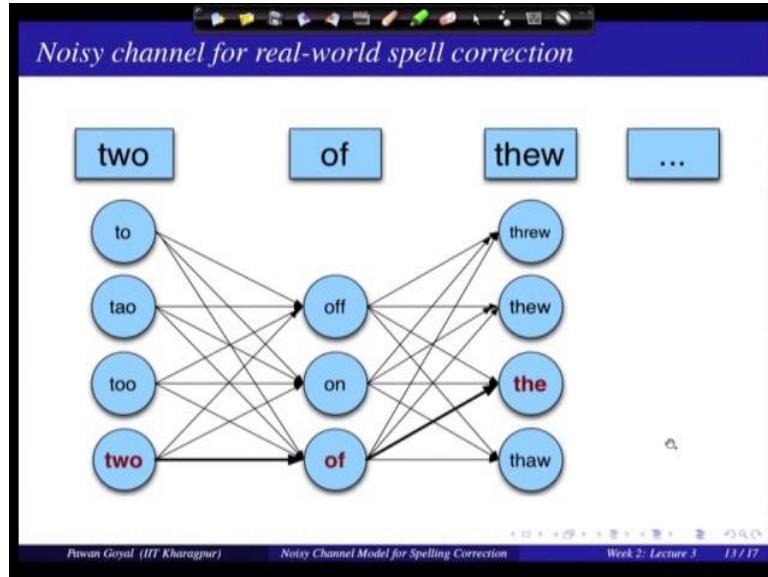
- Candidate ( $w_1$ ) =  $\{w_1, w'_1, w''_1, w'''_1, \dots\}$
- Candidate ( $w_2$ ) =  $\{w_2, w'_2, w''_2, w'''_2, \dots\}$
- Candidate ( $w_3$ ) =  $\{w_3, w'_3, w''_3, w'''_3, \dots\}$

Below the list, a red box highlights the first two items of the list. At the bottom of the purple box, there is a line of text: "Choose the sequence  $W$  that maximizes  $P(W|X)$ ".

Now suppose I have a sentence  $X$  that contains words  $W_1$  to  $W_n$  now we do not know which of these words is in error because all of these are real words in my dictionary. So, how do I actually start doing the spell correction? So, what I will do for each of the individual word I will try to find out what are the best possible candidates that might be the correct words and there I should not forget to include this word itself because this might be a correct word also. So, here what we are doing for the word  $W_1$ , we are saying candidates are  $W_1$  or some word  $W_1'$   $W_1''$   $W_1'''$  and so on, they are possible candidates for correction.  $W_2$  again that is what  $W_2$  and other candidates same way  $W_3$ , now what is my problem? Now I can have any sequence from the candidates of  $W_1$ , I can choose any one  $W_2$ , I can choose any one and so on and among all these possible sequences I have to find out which one is giving me the highest probability in something.

I want to find the sequence  $W$  that maximizes the probability of  $W$  given  $X$  that is whether this would be a good sequence or this would be a good sequence or this would be a good sequence or so on, which of this sequence will be the best in this case. So, again you have to find out  $W$  that maximizes probability of  $W$  given  $X$ . So, again can we try to use the noisy channel model? So, we will see.

(Refer Slide Time: 27:15)



Now we are taking a real example here 2 of w, thew and probably it should have been 2 of the; and we are not seeing the other words. So, as my; the model what will I do I will take the word two, I will put this word along with all the other possible candidates.

I have put in put t W o, t a o, t u, all are having a smaller distance similarly with off I am putting o n, o double f with the thew, I put thew, threw, thaw, the and so on and we know the correct sequences 2 off the fine, but in general suppose I had to find out the probability for each sequence this might become exponential. So, you can do the simple math suppose each word on an average has four possible candidates including itself and there are five different words in my sentence. So, how many different sequences I will have? 4 to the power 5 and if the number of words increases in my sentence this will keep on increasing.

(Refer Slide Time: 28:31)

Simplification: One error per sentence

Choose among all possible sentences with one word replaced

two of them?

- $w_1, w''_2, w_3$  two off them
- $w_1, w_2, w'_3$  two of the
- $w'''_1, w_2, w_3$  too of them

Pawan Goyal (IIT Kharagpur) Noisy Channel Model for Spelling Correction Week 2: Lecture 3 14/17 11:15 AM 10/10/2016

Now to avoid this problem, we will try to use some sort of we make some assumptions that it is also simplification to the model what is that. We say let us assume there will be at most one error in the sentence. So, how does it help? It helps in that whenever I am choosing the candidates, I will only choose a different word for one of the words. So, what do I mean by that? Suppose I had my sentence  $W_1, W_2, W_3$  and this has some other possible things like  $W_2'$ ,  $W_2''$ ,  $W_1'$ ,  $W_1''$ ,  $W_3'$ ,  $W_3''$ . So, in the previous model, we would have taken all the 3 cube that is 27 different candidates in the previous one.

So, with this assumption what we will do? For  $W_1$ , if I am choosing a candidate of  $W_1'$ , I will assume  $W_2$  and  $W_3$  were correct. So, this is the assumption there might be at most one error per sentence.

How many candidates will be there? I will have 2 possibilities here, 2 possibilities here and 2 possibilities here. So, there would be at most 6 plus 1, if you are taking the 1, there the original sentence might have been correct say  $W_1, W_2, W_3$  as it is. So, the candidates are  $W_1', W_2, W_3$  only one error per sentence  $W_1'', W_2, W_3$  and so on. So, this hugely reduces the number of candidates for which I have to check and this is quite true also and that is why you do not make more than one error per sentence in general.

(Refer Slide Time: 30:18)

*Simplification: One error per sentence*

Choose among all possible sentences with one word replaced

two of thew

- $w_1, w''_2, w_3$  two off thew
- $w_1, w_2, w'_3$  two of the
- $w'''_1, w_2, w_3$  too of thew

Choose the sequence  $W$  that maximizes  $P(W|X)$

Pawan Goyal (IIT Kharagpur) Noisy Channel Model for Spelling Correction Week 2: Lecture 3 14 / 17

Now once we have this assumption we want to find this sequence  $W$  that maximizes the probability of  $W$  given  $X$ . So, I can again try to use my noisy channel model.

(Refer Slide Time: 30:33)

*Getting the probability values*

Noisy Channel

$$\hat{W} = \arg \max_{W \in S} P(W|X)$$

where  $X$  is the observed sentence and  $S$  is the set of all the possible sequences from the candidate set

$$= \arg \max_{W \in S} P(X|W)P(W)$$

$$P(X|W) = \frac{P(w_1|w)}{P(w_1|w)}$$

Handwritten note:  $P(w_1|w) = p(w'_1|w) p(w_2|w_1) p(w_3|w_2)$

Pawan Goyal (IIT Kharagpur) Noisy Channel Model for Spelling Correction Week 2: Lecture 3 15 / 17

How find out the sequence  $\hat{W}$  that maximizes the probability of  $P(W|X)$ ,  $X$  is my observed sequence and  $W$  is one of my candidate sequences and we know how to find out the candidate sequences. So, that is for each of these sequence is  $W$ , I have to find out the probability  $W$  given  $X$  now as for the noisy channel model how will I write it down. So, I will again use the Bayes theorem here because I am starting from  $W$  and going to  $x$ . So, I can only

find out the probability of X given W. So, this is what I will do now how do I write probability X given W X is a sequence W is a sequence. So, suppose they have the same number of words I can make a simplifying assumption that P X given W is simply multiplication of probability of individual word given that in the sequence.

So, suppose my word is this is my W and this is my x. So, probability X given W i will write X probability of W 1 prime given W 1 probability W 2 given W 2 probability W 3 given W 3 . So, now, there are 2 kind of probabilities here one is when the 2 words are not the same probability of doing an error yes one they at the same now how do I find out the individual probabilities this one we have already see in the previous case, I will find out what are the [audit/edit] edit operation and what is the corresponding probability if it is deletion insertion and so on.

This is same as the previous case same as in previous, but how do I find out the probability of W 2 given W 2 should it be 1, now it will depend on what is the kind of errors that you are seeing in your corpus if you are seeing that they are on an average one error per 10 words, you will say this probability is 0.9, thus this is what is correct these are probability of the same; the word is correct.

If there are 1 error in 100 words you will say that this probability is 0.99 and that is how you will fix this probability.

(Refer Slide Time: 33:02)

*Getting the probability values*

*Noisy Channel*

$$\hat{W} = \arg \max_{W \in S} P(W|X)$$

where  $X$  is the observed sentence and  $S$  is the set of all the possible sequences from the candidate set

$$= \arg \max_{W \in S} P(X|W)P(W)$$

*P(X|W)*

- Same as for non-word spelling correction
- Also require probability for no error  $P(w|w)$

Ptwan Goyal (IIT Kharagpur)      Noisy Channel Model for Spelling Correction      Week 2: Lecture 3      15 / 17

(Refer Slide Time: 33:11)

Probability of no error

What is the probability for a correctly typed word?  $P(\text{"the"}|\text{"the"})$

*It may depend on the source text under consideration*

- 1 error in 10 words → 0.9
- 1 error in 100 words → 0.99

Pawan Goyal (IIT Kharagpur) Noisy Channel Model for Spelling Correction Week 2: Lecture 3 16 / 17

$P(X \text{ given } W)$  same as non word spelling correction, but we also need this probability  $w$  given  $w$  and that is the probability that you have correctly typed a word and that you can find by the kind of the amount of errors you are seeing in your source. If there is one error per 10 words you will say this probability is 0.9, if there is one error per 100 words you will say this probability is 0.99 that is how you will set all these probabilities and you will choose the candidate  $X$  that gives you the maximum probability, candidate  $W$  sorry.

(Refer Slide Time: 33:37)

Computing  $P(W)$

$W = w_1 w_2 w_3 \rightarrow$   
Language Model  $\rightarrow$

*Use Language Model*

- Unigram ✓
- Bigram ✓
- ...

Pawan Goyal (IIT Kharagpur) Noisy Channel Model for Spelling Correction Week 2: Lecture 3 17 / 17  
13:10 AM 10/10/2018

Now, the other part is finding out probability  $W$  by  $W$  I mean the sequence  $W_1$  prime,  $W_2$ ,  $W_3$  how do I define the probability of the sequence  $W$ . For that we will use a technique called language modeling. So, that will give me the probability of  $n$  is the sequence of words and we will use any of this unigram model bigram model or any other model for finding out this probability of  $W$ .

This here will be the topic for the next lecture. So, in this lecture we discussed what are non word errors, re-word errors how do we use noisy channel model to correct those, but there we also need something like probability of the sentence as such probability of the sequence of words  $W$ , how do we obtain that and that is why we will go to the idea of language modeling in the next lecture.

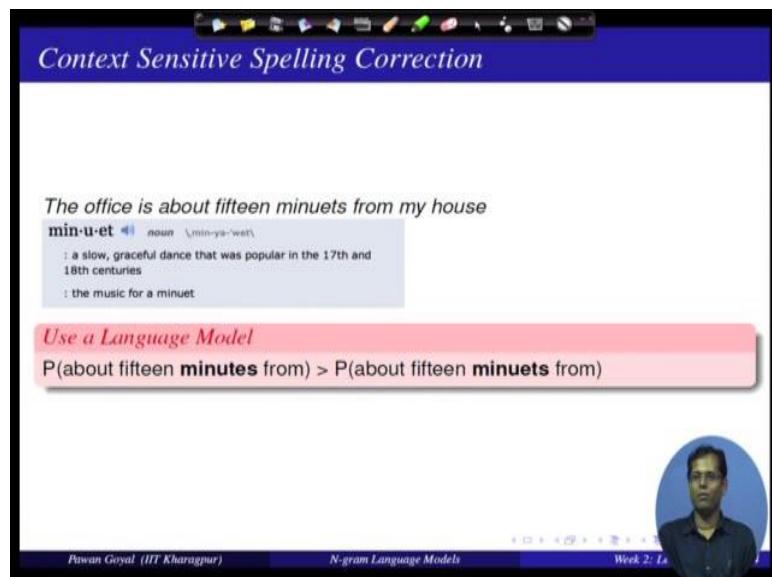
**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute Technology, Kharagpur**

**Lecture – 09**  
**N-Gram Language Models**

Welcome to the forth lecture of second week. So in the last lecture we were discussing about a spelling correction and also seeing how do we do that it using the contexts. And remember the last problem that we are facing is how do you find the probability of sequence of words. So I have sequence of words in my sentence that might be candidate what the probability of that sequence. And that is why we said we will be using language models for it. So today the talk, that the topic of this lecture is to see what are my N-gram language models and we will start with some motivation of what are the different applications where we might need them. So N-gram language modeling is a very nice technique in NLP and it is applied in many different applications. It is very one of the very basic concepts in NLP. Let us see what are my N-gram language models.

So I will start with the motivation of context sensitive spelling correction that was a topic of just the previous lecture.

(Refer Slide Time: 01:29)



Suppose I am having the sentence the office is about 15 minuets from my house. So here you can clearly see that this is just a spelling error minutes has been typed as minuets,

but suppose the word in minuets is also in my vocabulary. Suppose I see my vocabulary and I find that minuet is some sort of slow graceful dance that was popular in 17th and 18th centuries. So this word in my vocabulary, now, that means, I cannot easily detect that this is the incorrect word by using some isolated word as correction. So I should be using the context for that. So what is the probability that, so I will try to use some sort of language model to say with a probability of this a trends about 15 minutes from it is higher than the probability of a trends about 15 minuets from.

If you observe these 2 a trends you can yourself see that for it will should be more probable, but how do you formally defined these probabilities. And what is the model that we use for it. And that is what we will be seeing in this language model.

(Refer Slide Time: 02:47)

Probabilistic Language Models: Applications

**Speech Recognition**

- $P(\text{I saw a van}) >> P(\text{eyes awe of an})$

**Machine Translation**

Which sentence is more plausible in the target language?

- $P(\text{high winds}) > P(\text{large winds})$

**Other Applications**

- Context Sensitive Spelling Correction
- Natural Language Generation
- ...

Pawan Goyal (IIT Kharagpur) N-gram Language Models Week 2: La

So what are some of the other applications for this is helpful? Remember one of the earlier slides in our last week you were saying, in this case of speech recognition you face this problem that when you are uttering something you have to transcribe that. So whenever I am saying I saw a van it might also sound like eyes awe of n. So among the 2 a trends is which one is more likely. So again can I give some sort of probability value to each of these sequence in the say that probabilities I saw a van is more probable than the other.

Similarly, in machine translation there is a problem of collocations. That certain words even if they are correct translations do not occur much in the language with a in the

context of others like if I say if I have the word winds and before that as an adjective I can put either high or large. So for example, I have an Hindi (Refer Time: 03:48) [F] and I want to translate that into English – [FL] will be winds. Now for [FL] it might happened that both translations high and larger possible, in English. Now among those which one should I choose? And suppose from my corpus again find this language knowledge that high winds occur with more probability than large winds then I would say that high is a more probability translation than large.

And there are other applications is spelling correction. We have seen and you might have to use that for generation. Whenever you have to generate sentences again which particular generation has the higher probability?

(Refer Slide Time: 04:31)

Completion Prediction

- Language model also supports predicting the completion of a sentence.
  - ▶ Please turn off your cell ...
  - ▶ Your program does not ...
- Predictive text input systems can guess what you are typing and give choices on how to complete it.

Pawan Goyal (IIT Kharagpur) N-gram Language Models Week 2: La

Another possible application that you might be using it in say google search or other places whenever you are trying to type something that they will be predict what will be the next word. So whenever I am saying please turn off your cell what will be the next one you can bit probably phone. Please turn off your cell phone. And your program does not probably compared or something. So given in a sequence of words how they predict what is the next for that is going to come in this sentence. So that is my auto completion task. That is huge a lot in a coyote completion or even when you are typing something in your SMS or (Refer Time: 05:08) you might have certain softwares that do that. So they also use the concept of language modeling.

So these are also called the predictive text input systems. So given whatever you are typing they will give you choices in how do you complete it. What does the various possibilities were, which will complete?

(Refer Slide Time: 05:32)

**Goal:** Compute the probability of a sentence or sequence of words:  
 $P(W) = P(w_1, w_2, w_3, \dots, w_n)$

**Related Task:** probability of an upcoming word:  
 $P(w_4 | w_1, w_2, w_3)$

A model that computes either of these is called a **language model**.

Pawan Goyal (IIT Kharagpur)      N-gram Language Models      Week 2: L1

So given these applications in background, let us see what is my language model what is the goal what do we gone to achieve. So we talked about this. In the previous lecture we said we I want to complete probability of w that is what is the probability of the sequence w 1 to w n. This is the sequence of words what is the probability of the sequence. So this is one of the gold. Compute the probability of a sentence or a sequence of words, P w. What is the other related task that we saw in prediction system that given 3 words w 1, w 2, w 3 what you will be my w4? So probability of w 4

$$P(w) = P(w_1, w_2, w_3, \dots, w_n)$$

given the 3 previous words, probability of an upcoming words .

$$P(w_4 | w_1, w_2, w_3)$$

Now, in general any model that computes either of these probability of the sequence or probability of the word given the previous a trend is called a language model. We were see language model using these definitions.

(Refer Slide Time: 06:35)

The slide has a blue header bar with the title "Computing P(W)". Below the header, there is a purple box containing the text "How to compute the joint probability" and "P(about, fifteen, minutes, from)". A pink box below it contains the text "Basic Idea" and "Rely on the Chain Rule of Probability". At the bottom left, it says "Pawan Goyal (IIT Kharagpur)" and "N-gram Language Models". On the right, it says "Week 2: Le" and shows a small circular profile picture of a man. The bottom of the slide features a standard Windows taskbar.

So now how do I actually compute probability of the sequence, w means sequence of words here. So suppose from the examples that we were taking initially I have the sequence about 15 minutes from. I want to compute the probability of the sequence. Now suppose I do not tell you anything else what is the simplest model that you will apply to compute this probability. So you will probably apply the chain rule of probability. Yes. So remember chain rule of probability.

(Refer Slide Time: 07:12)

The slide has a blue header bar with the title "The Chain Rule". Below the header, there is a purple box containing the text "Conditional Probabilities" and the formula  $P(B|A) = \frac{P(A, B)}{P(A)}$ . Below the formula, there are two red handwritten equations:  $P(A, B) = P(B|A) P(A)$  and  $P(A, B, C) = P(A) P(B|A) P(C|A, B)$ . At the bottom left, it says "Pawan Goyal (IIT Kharagpur)" and "N-gram Language Models". On the right, it says "Week 2: Le" and shows a small circular profile picture of a man. The bottom of the slide features a standard Windows taskbar.

So it derives from conditional probabilities. So your conditional probabilities you might remember. So that is what is the probability of B given A that is probability of the joint. The joint probability P A B divided by probability A . Now you

$$P(B|A) = P(A, B)/P(A)$$

can also use that to writing in some other way. So you can say probability A B which probability B given a times probability A . And if you can do

$$P(A, B) = P(B/A). P(A)$$

that for any number of words in my sentence or in general any number of events in probability. So you can say probability A B C which probability A probability B given a probability C given A B and you can keep on

$$P(A, B, C) = P(A). P(B/A). P(C/AB)$$

doing that finite numbers and that is my chain rule of probability.

(Refer Slide Time: 08:11)

The slide is titled "The Chain Rule". It contains three main sections:

- Conditional Probabilities**: Shows the formula  $P(B|A) = \frac{P(A, B)}{P(A)}$  and the factorized form  $P(A, B) = P(A)P(B|A)$ .
- More Variables**: Shows the formula  $P(A, B, C, D) = P(A)P(B|A)P(C|A, B)P(D|A, B, C)$ .
- The Chain Rule in General**: Shows the formula  $P(x_1, x_2, \dots, x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2)\dots P(x_n|x_1, \dots, x_{n-1})$ .

At the bottom, there are navigation icons and text: "Pawan Goyal (IIT Kharagpur)", "N-gram Language Models", "Week 2: Lecture 4", and "7 / 24".

I can write P A B A P A times P B given A . In general, even if

$$P(B|A) = P(A, B)/P(A)$$

I am more variables I can do the same. Same chain rule of probabilities idea and this is the general formulation probability x 1 given x 1 is probability x 1 probability x 2 given x 1 and so on and probability x 1 given x 1 to x n minus 1.

So now given this chain rule now go back to an initial problem. Probability of about 15 minutes from, so how do I write it using the chain rule of probabilities?

(Refer Slide Time: 08:49)

$P(w_1 w_2 \dots w_n) = \prod_i P(w_i | w_1 w_2 \dots w_{i-1})$

$P(\text{"about fifteen minutes from"}) =$   
 $P(\text{about}) \times P(\text{fifteen} | \text{about}) \times P(\text{minutes} | \text{about fifteen}) \times P(\text{from} | \text{about fifteen minutes})$

Pawan Goyal (IIT Kharagpur)      N-gram Language Models      Week 2: 1a

So how will you right this probability? You will say this is same as probability of about times probability of 15 given about times probability minutes given about 15 probabilities from given about 15 minutes. So you can right it like that. Yes, now suppose I even increase this to about 15 minutes from office what will you do.

(Refer Slide Time: 09:15)

*Count and divide*

$P(\text{office} | \text{about fifteen minutes from}) = \frac{\text{Count}(\text{about fifteen minutes from office})}{\text{Count}(\text{about fifteen minutes from})}$

*What is the problem*

We may never see enough data for estimating these

Pawan Goyal (IIT Kharagpur)      N-gram Language Models      Week 2: 1a

And you will write it in terms of this probability of this given about 15 minutes from. So now, how the question that comes is, find, I can write it in terms of chain probability chain rules. How do I actually compute these probabilities? Now what is probability of

office given about 15 minutes from that would be in my corpus how many times do I observe about 15 minutes from. Among those how many times I observe office after that. Now what is one particular proper you will face in doing it in this work? You might not have seen sufficient number of about 15 minutes from in the corpus. Suppose it occurred only once it occur with the office. This probability will be one, but does not mean that only office can occur this about this occurrence. And this problem becomes severe as you keep on going to the higher and higher number of words in the conditional.

So we will never see enough data for estimating these. So I cannot estimate easily what is the probability of office given about 15 minutes from so that means, this will not work as it is. So we will need to do certain simplifications.

(Refer Slide Time: 10:32)

*Simplifying Assumption: Use only the previous word*

$$P(\text{office} \mid \text{about fifteen minutes from}) \approx P(\text{office} \mid \text{from})$$

*Or the couple previous words*

$$P(\text{office} \mid \text{about fifteen minutes from}) \approx P(\text{office} \mid \text{minutes from})$$

So what is the simplifying assumptions that we make? We say we can probably use only the previous word and forget about everything else. So I have to compute the probability of office given about 15 minutes from and it is; that is approximated by probability of office given from. I forget the other terms. So or I can use the previous 2 words. So this can be written as the probability of office given minutes in from. So this is simplification that we are making. Why we are doing that? Because now it is easy to find complete the probabilities. What are the words that come of that from and how what session of times office comes after from, but this was not possible when I was taking condition on 4

previous words? Because information was probably not seeing them enough in my data, so this helps us giving a better estimate of these probabilities.

(Refer Slide Time: 11:32)

More formally we can use  $k$ th order Markov model. So I have this information have to

$$P(w_1, w_2, \dots, w_n) = \prod_i P(w_i | w_{i-k}, \dots, w_{i-1})$$

compute the probability of the sequence of  $w_1$  to  $w_n$ . By using chain rule I write it as this  $w_i$  given the

$$P(w_1, w_2, \dots, w_n) = \prod_i P(w_i | w_1, w_2, \dots, w_{i-1})$$

previous  $i$  minus 1 words multiply for all the words. Now in  $k$ th order Markova model assumption what I will do? I conditioned it only on the previous  $k$  words. So I will write probability  $w_i$  given  $w_1 w_2 \dots w_{i-1}$  as  $w_i$  given only the previous  $k$  words. So here you are using only 1 up to  $k$  previous words not all the  $i$  minus 1 words. So this is  $k$ th order Markov model assumption.

(Refer Slide Time: 12:35)

*P(office | about fifteen minutes from)*

An  $N$ -gram model uses only  $N - 1$  words of prior context.

- Unigram:  $P(\text{office})$
- Bigram:  $P(\text{office} \mid \text{from})$
- Trigram:  $P(\text{office} \mid \text{minutes from})$

*Markov model and Language Model*

An  $N$ -gram model is an  $N - 1$ -order Markov Model

So we can do that for all the come all the components in this product. Now and that is how we will define ever N-gram models. So if I take this particular probability office given about 15 minutes from, if I am using only the previous word that will be using it 2-gram language model. So in general if I am using only  $n$  minus 1 words of prior context I am defining in N-gram language model. So here if I am using if I am not using any word from the context 0 words from the context this is a unigram language model,  $n$  is equal to 1 here. If I am using 1 word from the context, then it is a bigram language model,  $n$  is equal to 2. If I am taking 2 words from the context it is a trigram language model,  $n$  is equal to 3. Now can you try to relate in N-gram language model with some  $k$ th order Markova assumption?

In  $k$ th order Markova assumptions; you were using  $k$ th previous word. So if I am using  $k$  previous words I have a  $k$  plus 1-gram language model. So I can say an N-gram language model is an  $n$  minus one  $h$  order Markova model. In N-gram language model I use  $n$  minus 1, words from the context. So this is the relation between an N-gram language model. And an  $n$  minus 1 order Markova model.

(Refer Slide Time: 14:06)

The slide has a blue header bar with the title "N-Gram Models". The main content area contains the following bullet points:

- We can extend to trigrams, 4-grams, 5-grams
- In general, an insufficient model of language:  
*language has long-distance dependencies:*  
"The computer which I had just put into the machine room on the fifth floor **crashed**."
- In most of the applications, we can get away with N-gram models

At the bottom of the slide, there is a footer bar with the text "Pawan Goyal (IIT Kharagpur)", "N-gram Language Models", and "Week 2: Lec 1". To the right of the footer, there is a small circular video player showing a person speaking.

Now internal we can extended to trigrams we assume 2 words from the context. 4 grams 3 words from the context, of 5 grams 4 words from the context, but in general we cannot see that any N-gram will be sufficient model for the language. Why? Because language we also see some long term dependencies. So consider this sentence. The computer which I had just put into the machine room on the fifth floor crashed.

Now, what is the word on these the word crashed depends? If you were see here the word crashed depends on the about computer, but this cannot be captured by using 2 gram, 3 gram, 4 gram, 5 gram, 6 gram, so on to see. So you might have to go to 11 12 grams and that is probably not very advisable. So you use must always knew that any N-gram model is not a sufficient model of a language, but it captures many word ordering relative regularities. So in most of the applications we can use simple N-gram model with an, it will 2 3 and we will get up with it.

(Refer Slide Time: 15:19)

*Maximum Likelihood Estimate*  
Value that makes the observed data the "most probable"

$$P(w_i|w_{i-1}) = \frac{\text{count}(w_{i-1}, w_i)}{\text{count}(w_{i-1})}$$
$$P(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

Pawan Goyal (IIT Kharagpur)      N-gram Language Models      Week 2: Lec

So now the question is how do we estimate these N-gram probabilities from a corpus. So for that the simple method of estimating these probabilities is by using some maximum likelihood estimate. So what is that? Suppose I have to compute the probability of  $w_i$  given  $w_{i-1}$ . So I will find out in my corpus how many times the word  $w_{i-1}$  occurred. Among those what fraction of times  $w_i$  occurs after that

) / .

$$P(w_i|w_{i-1}) = \text{count}(w_{i-1}, w_i) / \text{count}(w_{i-1}) \text{ or } P(w_i|w_{i-1}) = c(w_{i-1}, w_i) / c(w_{i-1})$$

Remember we were doing it from the one of the earlier slides in versatile actress whose. So how many times the word actress comes after versatile what fraction of times? Same here what is the fraction of times that the word  $w_i$  occurs after the  $w_{i-1}$ , which defines a probability distribution. After  $w_{i-1}$  each word in my vocabulary can come and this will be a probability distribution.

So in general I can give a notation of  $c$  instead of  $\text{count}$   $c$   $w_i$   $m_i$  means number of times  $w_i$  occurred in my corpus and so on.

(Refer Slide Time: 16:31)

An Example

$$P(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

<s>I am here </s>  
<s>who am I </s>  
<s>I would like to know </s>

*Estimating bigrams*

$P(I|<s>) = 2/3$   
 $P(</s>|here) = 1$   
 $P(would | I) = 1/3$   
 $P(here | am) = 1/2$   
 $P(know | like) = 0$

Pawan Goyal (IIT Kharagpur) N-gram Language Models Week 2: Lec 1

Now, let us take examples and see how do I estimate these probabilities. So I have 3 sentences in this corpus. I am here, who am I and I would like to know. And you also have some tokens on a start of the sentence and the end of the sentence. And you want to compute the bi bigram probability  $w_i$  given  $w_{i-1}$ . So how will you do that? So suppose I want to compute all these probabilities. Probability of  $i$  coming at the start of the sentence; probability of  $here$  coming with this probability end of the sentence coming after  $here$  probability of  $would$  coming after again so on. So how do I compute these probabilities? So I just take the first one probability of  $i$  coming at the start of the sentence.

So I will find out how many of start of the sentences I have seen. Among those perfection of the times  $i$  occurred. So I have seen 3 start of the sentence, out of those twice highly occurred. So this probability would be 2 by 3. Here how many times I have seen  $here$ , one and among those how many times the end of the sentence occurred also one this will be one by one and so on.

So this is what I find 2 by 3 1; 1 by 3 1 by 2 and 0. So this is easy given a corpus you can find out the bi gram probabilities by it is just giving the counts.

(Refer Slide Time: 18:10)

The slide title is "Bigram counts from 9222 Restaurant Sentences". Below the title is a table showing bigram counts. The table has columns for each word and rows for each word. The counts are as follows:

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

Pawan Goyal (IIT Kharagpur)      N-gram Language Models      Week 2: Lec 1

Now, this is some example from some restaurant corpus. So that had 9000 plus sentences and these some bigrams that were the numbers are given here. So can you see some regularities here. So i and i do not occur together much, but i and want occurred 827 times. So you are in a restaurant after I mostly say want I want some sort of food. So I want is a bigram that occurs a lot in this data. What are the others bigrams that occurs a lot, want to I want to do something? Eat Chinese, Chinese food. Eat lunch, to spend so all these bigrams that occurs a lot. So this tells in lot about that corpus. So I am talking about the restaurant corpus where it is more about ordering some food and eating some, so a kind of foods.

(Refer Slide Time: 19:09)

The slide has a blue header bar with the title "Computing bigram probabilities". Below it is a purple box labeled "Normalize by unigrams" containing a table:

	i	want	to	eat	chinese	food	lunch	spend
i	2533	927	2417	746	158	1093	341	278

Below this is another purple box labeled "Bigram Probabilities" containing a table:

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

At the bottom left is the text "Pawan Goyal (IIT Kharagpur)", in the center "N-gram Language Models", and at the bottom right "Week 2: Lecture 1". A small video player in the bottom right corner shows a person speaking.

Now, suppose I have to compute the probability, bigram probability. What is do I need? I need the count number of times this were occurs without the word and you know divided by the unigram probability of the previous word. So suppose I give you the unigram counts also of all these words can you come with the bigram probabilities? Yes, it is becoming quite easy. So now, I just divided by the number of times i occurs. So number of times i amount of the together divided that number of times i occurs. So this gives you probability goes to 0.33. So probability of want occurring after i is 0.32 from this corpus and so on. I can compute all the words in this corpus.

(Refer Slide Time: 19:52)

The slide has a blue header bar with the title "Computing Sentence Probabilities". Below it is a purple box containing a mathematical expression:

$$P(<\text{s}> | \text{I want english food} </\text{s}>) = P(\text{I} | <\text{s}>) \times P(\text{want} | \text{I}) \times P(\text{english} | \text{want}) \times P(\text{food} | \text{english}) \times P(</\text{s}> | \text{food}) = 0.000031$$

At the bottom left is the text "Pawan Goyal (IIT Kharagpur)", in the center "N-gram Language Models", and at the bottom right "Week 2: Lecture 1". A small video player in the bottom right corner shows a person speaking.

So now given a corpus, you should be confident now that how do I compute the bigram probabilities. Now suppose you have computed bigrams probabilities from the corpus. Now can use that to solve over initial problem that is, how do I find the probability of this sentence this sequence of words. So I have the sentence here. A start of the sentence, I want English food and in end of the sentence. I want to find the probability of the sentence. So how do use bigram model do that. I will say this is probability of i given as the start of the sentence times probability want given i times probability English even want and so on, because I am actually using the chin rule of the probabilities, but I am using in first short of a Markova assumption. So that gives me all this probability and I multiply that and then it gives me the probability of this sentences each.

So if I use the previous probability that will be flip 0.000031. And now you can give me any sentence and I can use my bigram probabilities to find out the probability of the sentence. And I can solve all my problems of (Refer Time: 21:02) spelling corrections and other scenarios.

(Refer Slide Time: 21:09)

The slide has a blue header bar with the title "What knowledge does n-gram represent?". Below the title is a large white area containing a bulleted list of probability calculations. At the bottom right is a circular portrait of a man. The footer of the slide includes the name "Pawan Goyal (IIT Kharagpur)", the topic "N-gram Language Models", and the week "Week 2: Lec 1".

- $P(\text{english}|\text{want}) = .0011$
- $P(\text{chinese}|\text{want}) = .0065$
- $P(\text{to}|\text{want}) = .66$
- $P(\text{eat} | \text{to}) = .28$
- $P(\text{food} | \text{to}) = 0$
- $P(\text{want} | \text{spend}) = 0$
- $P(\text{i} | \text{<s>}) = .25$

So from this corpus what does the knowledge N-gram represent? So these are some values. So what do you try to infer from there. So probability English given want is 0.0011. And probability Chinese given want is 0.0065 what does it that tell. So if you see that will tell that the any Chinese food it is 6 time more popular than the English food in whatever from whatever detail it was taken from. To give want is 0.66, so that is 2

occurs a lot with a corpus after I want. I want to do something eat after to is again 0.28. Someone want to eating to do something into eat occurs a 0.28 probability. Food given to a 0 that means, the word food never occurs after to and that is something that talks about the language.

In language generally use a verb after to, I want to do something and here the food is a noun so this will not occur in my data. Similarly, want given is spend as a 0, again some fact about language that 2 words. Generally, do not occur occurred simultaneously. And I give the start of the sentence 0.25 again gives some in instigation that most of the sentences start with i in that in that corpus. So mostly about I want to do something. So you started with i. So these N-grams might represent some knowledge about the language and grammar as such or about the particular data or domain that you are trying to build it this form. And this is some idea that we can use for even modeling different domains in my data separately. I can build different language models for each domain, will talk about this further.

(Refer Slide Time: 23:04)

*Practical Issues*

*Everything in log space*

- Avoids underflow
- Adding is faster than multiplying

$$\log(p_1 \times p_2 \times p_3 \times p_4) = \log p_1 + \log p_2 + \log p_3 + \log p_4$$

*Handling zeros*

Use smoothing

Pawan Goyal (IIT Kharagpur)      N-gram Language Models      Week 2: Lec

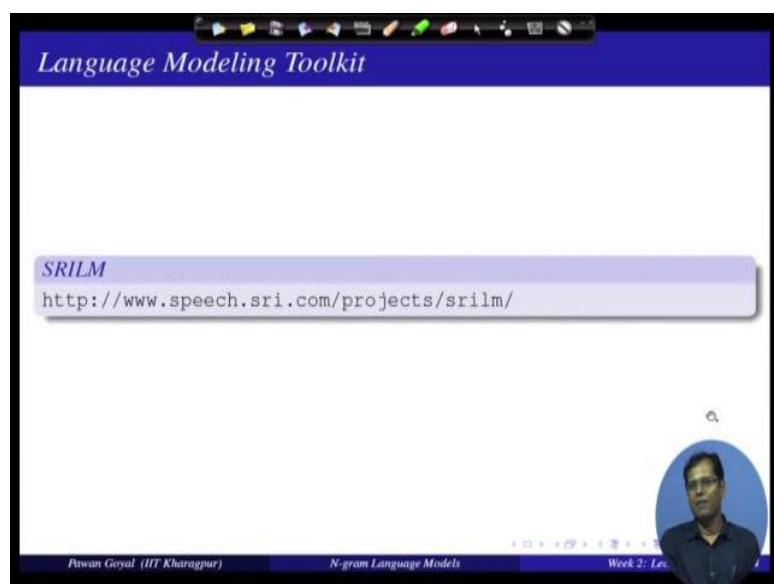
So there are certain practical issues that you might have to aware, of like when I am doing this probability computation for a sentence I am actually doing multiplication of many probability values. And all of these might be very small. So if I simply do multiplication of probabilities it might lead to some underflow and this might just end up in getting a 0 values for all the probabilities. So it is better that you do everything in log a

space in that way you are simply adding them and in any case adding each individual operation or more efficient operation than multiplication. So you are just storing the log of probabilities and you are adding these problem these logs. So you can if you aware to find out probabilities  $p_1$  times  $p_2$  times  $p_3$  times  $p_4$ , if you convert in log space that is nothing, but  $\log p_1$  plus  $\log p_2$  plus  $\log p_3$  plus and so on

$$\log(p_1 \times p_2 \times p_3 \times p_4) = \log p_1 + \log p_2 + \log p_3 + \log p_4$$

So you can restore a log values and you can just add this. There is another problem of handling zeros. Suppose in trends there is a particular bigram that you see that never occurred you a trend data. So what would happen? You will give it a value zero, but that will convert the whole, everything 0. So you need to do something for that, and we will see that in the concept of a smoothing.

(Refer Slide Time: 24:28)



So there are some popular toolkits are available. So SRILM is one popular toolkit, but there are many other toolkits that you can use for language modeling.

(Refer Slide Time: 21:40)

The screenshot shows a presentation slide with a blue header bar containing icons. The main title is "Google N-grams". Below the title, there is a list of statistics:

- Number of tokens: 1,024,908,267,229
- Number of sentences: 95,119,665,584
- Number of unigrams: 13,588,391
- Number of bigrams: 314,843,401
- Number of trigrams: 977,069,902
- Number of fourgrams: 1,313,818,354
- Number of fivegrams: 1,176,470,663

Below the statistics is a link: <http://googleresearch.blogspot.in/2006/08/all-our-n-gram-are-belong-to-you.html>.

At the bottom of the slide, there is a footer bar with the text "Pawan Goyal (IIT Kharagpur)", "N-gram Language Models", and "Week 2: Lec". On the right side of the footer, there is a small circular video player showing a man speaking.

You can also; if you want to use some large corpus data you can try to use these google N-grams. So there again available on this link and there you will find out for each different N-gram what is the number of times they occurred in the corpus.

So if you go to this link you will find out there are huge number of tokens sentences and all. They gave you unigrams bigrams trigrams 4 grams and 5 grams.

(Refer Slide Time: 25:12)

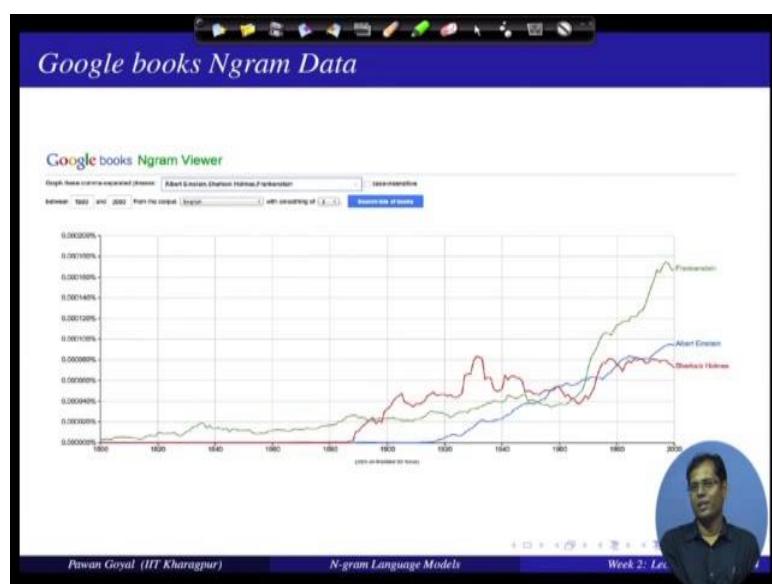
The screenshot shows a presentation slide with a blue header bar containing icons. The main title is "Example from the 4-gram data". Below the title, there is a list of 4-grams and their counts:

- serve as the inspector 66
- serve as the inspiration 1390
- serve as the installation 136
- serve as the institute 187
- serve as the institution 279
- serve as the institutional 461

At the bottom of the slide, there is a footer bar with the text "Pawan Goyal (IIT Kharagpur)", "N-gram Language Models", and "Week 2: Lec". On the right side of the footer, there is a small circular video player showing a man speaking.

So what are some of the examples? So suppose I am sing it after a starting from serve as the. So, that data will tell you serve as the inspector occurred 66 times in the corpus. Serve as the inspiration occurs 1390 times. Serve as the installation occurred 136 times and so on. So this is a fourgram that you find from the data, but you see these are only the counts. Now how will you use that the compute the probability? That probability will inspector given serve as the, for that you will also need to use the trigram count of serve as the; so again this 5 is available you can you can get this 4 gram of the data and trigram data and try to compute the all the probability values.

(Refer Slide Time: 25:56)



So they also give a nice API by which you can visualize many interesting patterns in the usage of words. So because heritage divides across many different centuries you can also plot it temporally. So what we are seeing here, suppose I give 3 different queries Albert Einstein Sherlock Holmes and Frankenstein on google N-gram viewer, it tells me over the years what is the probability of these bigrams. So can you see something interesting here? So Sherlock Holmes became popular around 1885 and so on. Before that is when nearly 0; Frankenstein probably there in the novels even before that. So even from the 1800 he finds the occurrence of Frankenstein that keeps on increasing even 2000, it is more it is much more than Albert Einstein and Sherlock Holmes.

Albert Einstein started getting popular in the data around 1970s. That is why the most of the discoveries happen, happened and it came into the books and at some point of time it

became more popular than even Sherlock Holmes. So you can do nice sort of analysis of what kind of words came into the language all my data in what times, and how the popularity changed over the years by using these kind of data.

So today we will we gave only some intuition and what is language model, how do we compute that, and what we can use it from? For there were some problems that we saw that how do we handle the zeros. So with 0 is the whole probability of trends will go to 0. Even if one of the components has a probability of 0, so how do we avoid that problem, how do we solve that problem? That is where the concept of the smoothing will come in picture and that is what we will cover in the next lecture.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 10**  
**Evaluation of Language Models, Basic Smoothing**

So, welcome back for the final lecture for second week. So, in the last lecture, we were discussing about the basic of language modeling; and there we just hinted about one problem about the zeros in language models. So, today in this lecture, we will discuss not only how to handle this zeros in language modeling's, but also how do you go about evaluating your language model that you have learned from some data. So, these are the two main topics for this lecture.

(Refer Slide Time: 00:49)

*Evaluating Language Model*

*Does it prefer good sentences to bad sentences?*  
Assign higher probability to real (or frequently observed) sentences than ungrammatical (or rarely observed) ones

*Training and Test Corpora*

- Parameters of the model are trained on a large corpus of text, called **training set**.
- Performance is tested on a disjoint (held-out) **test data** using an **evaluation metric**

Pawan Goyal (IIT Kharagpur)      Evaluation of Language Models, Basic Smoothing      Week 2: Lecture 5      2 / 16

So, we start with the evaluation of language models. So, what is the basic criteria for evaluating a language model? So, one symbol intuition could be a language model is better if it is assigning a high probability to the actual sentences, the real sentences and a low probability sentences that are not so grammatical that can be one criteria for evaluating language models. But this has to be done manually. So, you will have to check what is a probability that your model is assigning to various sentences that are real, most of the sentences that are not real that are not grammatical, but can we do that in an automatic

manner. So, for that we may also use some sort of training and test data. So, what is the idea behind using this training and test data?

So, I have some corpus and I learn my language model on some part of that. So, this is my training set. Now, once I have learned that language model from my training set, I will test whether it is providing a good high probability for my test data. And I will have some evaluation metric for finding out how good this model is doing on my test data, and by this method I can even compare across different language models. So, if I have learned three different language models for example, I can find out which one does better on my test data, and that will be the best model. So, in general what are the different ways in which language models can be evaluated. So, as such there are two different criteria for evaluation, one is called extrinsic, another one is called intrinsic. We will discuss what are these individually.

So, what do I mean by extrinsic evaluation of my language models? So, suppose I have learned two different language models a and b from some training data. Now, what do we do in extrinsic evaluation, we try to use them on certain task. So, remember some of the applications that we have discussed of language models, so they can use in the spelling correction or in speech recognition and all that. So, now, what I will do in this extrinsic evaluation is that once I have learnt two different models, I will try apply them to these tasks individually, and then see what is the performance that I am obtaining for each of this task by different models. Now, I will say whichever model is giving a better performance on these task is my preferable model. So, this is called extrinsic evaluation or task based evaluation.

(Refer Slide Time: 03:32)

Extrinsic evaluation of  $N$ -grams models

Comparison of two models, A and B

- Use each model for one or more tasks: spelling corrector, speech recognizer, machine translation
- Get accuracy values for A and B
- Compare accuracy for A and B

Ptwan Goyal (IIT Kharagpur)      Evaluation of Language Models, Basic Smoothing      Week 2: Lecture 5      3 / 16

So, here we have discussed if we have mentioned three different tasks a spelling correction, speech recognition and machine translation. I will have two models A and B, I will get accuracy values for these three tasks and I will compare the accuracy and tell which language model is preferable than other. So, this is my extrinsic evaluation but is there some intrinsic way of evaluation without actually applying it on some task.

(Refer Slide Time: 03:57)

Intrinsic evaluation: Perplexity

Intuition: The Shannon Game

How well can we predict the next word?

- I always order pizza with cheese and ...
- The president of India is ...
- I wrote a ...

Unigram model doesn't work for this game.

A better model of text

is one which assigns a higher probability to the actual word

Ptwan Goyal (IIT Kharagpur)      Evaluation of Language Models, Basic Smoothing      Week 2: Lecture 5      4 / 16

So, for that so we have the notion of perplexity, so this is how we evaluate language model for intrinsic evaluation. So, intuition comes from the Shannon game if you heard about this

name. So, what is that game how well can you predict the next word given certain context. So, let us see some examples. So, I have the first sentences here, I always order pizza with cheese and there is a word that you have to predict may be you can say pepper or whatever. And similarly, the second sentence, the president of India is and you can predict the next word. Similarly, in the third sentence, I wrote a and you can predict letter or program or whatever.

So, now in my data I know what are the words that will fill up these blanks. Now, suppose I have two different language models, I will find out which of these fills up these blanks better than the other one. So, whichever one is good at predicting the next word is my preferable language models. So, one thing you can easily say from here, the unigram models are probably not built for this task. So, suppose you want to predict the next word using a unigram model, what would happen? So, if you remember unigram models do not use the context at all. So, you will just end up providing at every place the word that is having the highest probability. So, probably unigram models are not good for this task, but you can apply a bigram models the model which uses the previous word, trigram models that uses the previous two words and so on. So, finally, among different language models a better model is one that you will assign a higher probability to the actual word.

(Refer Slide Time: 05:54)

The best language model is one that best predicts an unseen test set

**Perplexity ( $PP(W)$ )**

Perplexity is the inverse probability of the test data, normalized by the number of words:

$$PP(W) = P(w_1 w_2 \dots w_N)^{-\frac{1}{N}}$$

**Applying chain Rule**

$$PP(W) = \left( \prod P(w_i | w_1 \dots w_{i-1}) \right)^{\frac{1}{N}}$$

**For bigrams**

$$PP(W) = \left( \prod P(w_i | w_{i-1}) \right)^{\frac{1}{N}}$$

Pawan Goyal (IIT Kharagpur)      Evaluation of Language Models, Basic Smoothing      Week 2: Lecture 5      5 / 16

Now, what is the formal definition of perplexity? So, in general, what we are trying to find out the best language model that predicts an unseen test data. So, how do I define perplexity?

So, this is a simple definition of perplexity. I have some test data; I am finding out the inverse probability of test data normalized by the number of words that I am seeing in my test data, that means finding the probability of test data and then normalize it with respect to the number of words and its inverse probability that means, if I have a low perplexity I have a better model. So, formally I can define my perplexity like that suppose in my test data I have words  $w_1$  to  $w_N$ , I am finding the probability of this whole sequence and normalizing it by minus 1 by  $N$  here

$$^{-1/N}.$$

$$PP(W) = P(w_1 w_2 \dots w_N)$$

So, now this is a general definition. Now, the next question that you might have is that where does the language model come into picture in this definition. The simple definition of finding probability of this whole address  $w_1$  to  $w_N$  and sub with some normalization. Now, where does the language model come into picture? Now, remember the chain rule that we discussed of probabilities if I have this sequence  $w_1$  to  $w_N$ , how can write the probability of the sequence in terms using the chain rule of probability, so that is what you will see here. So, I can write the same expression like that one divided by probability of  $w_i$  given the previous  $i - 1$  words this is in general this definition of probability  $w_1$  to  $w_N$

$$PP(W) = (\prod 1/P(w_i | w_1 \dots w_{i-1}))^{1/N}$$

Now, can you see how do we apply language models in this definition. So, in language model, we take one particular assumption about using this chain rule that is how many previous context I will be using. So, you can in place of this, you can replace any of the model that you have learned. So, suppose you want to replace with your bigram model that means, you want to find out what is the probability that the bigram model will give for this utterance. So, what you will do? So, we simply replace here the bigram model probability, so that will give you the perplexity for the bigram model that you have learned or trained using some training data. So, you will feed in all this probability  $w_i$  given  $w_{i-1}$  in that model and that will give you the perplexity of the bigram model. So, now just to give you an intuition what do I mean; what the number of number that I will get indicates perplexity value what does that indicate, let us take a simple example.

(Refer Slide Time: 08:33)

*Example: A Simple Scenario*

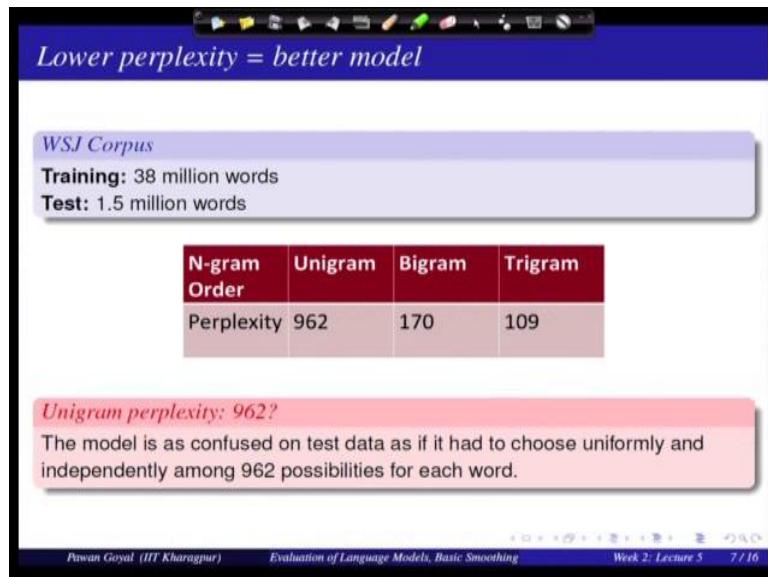
- Consider a sentence consisting of  $N$  random digits
- Find the perplexity of this sentence as per a model that assigns a probability  $p = 1/10$  to each digit.

$$\begin{aligned} PP(W) &= P(w_1 w_2 \dots w_N)^{-\frac{1}{N}} \\ &= \left( \left( \frac{1}{10} \right)^N \right)^{-\frac{1}{N}} \\ &= \left( \frac{1}{10} \right)^{-1} \\ &= 10 \end{aligned}$$

Pawan Goyal (IIT Kharagpur)      Evaluation of Language Models, Basic Smoothing      Week 2: Lecture 5      6 / 16

So, suppose I have a sentence that contains  $N$  random digits. And I have a model that assigns a probability of 1 by 10 to each digit. Now, I want to find a perplexity of the sentence using that model that will give me a probability of 1 by 10 to each digit. So, what will be the perplexity, let us try to use the definition. So, how did we define perplexity? That is probability of this whole sequence yes,  $w_1, w_N$  to the power minus 1 by  $N$ , yes. So, can you try to fill in the values? Suppose my model gives a probability of 1 by 10 to each word. So, this would be 1 by 10 to each word to the power  $N$ , for  $N$  different words  $N$  to the power minus 1 by  $N$ , so that is nothing, but 1 by 10 to the power minus 1 and that will give me 10. So, this tells me that the perplexity of this model is 10, the model that assigns a probability of 1 by 10 to each digit. Now, this might give you a hint of what this number indicates. So, let us take another example from some test data. So, what kind of perplexity values we get and how we can interpret this values.

(Refer Slide Time: 09:57)



So, an experiment was conducted over wall street journal corpus. So, in training they had 38 million words on which the language model was trained. And for testing they had 1.5 million words on which perplexity was computed. And they trained three different models unigram model, bigram model and trigram model. Now, and they found out what is the perplexity of the model in using the test set. So, these are the numbers that were found; for unigram model perplexity was 962, bigram 170, and trigram 109. So, now let us try to answer this question, what is this value of 962 perplexity in unigram indicate. So, what it means is that whenever my model is trying to assign a word, as if it has to choose among 962 different possibilities at each individual choice point independently and randomly.

So, this means the model is very, very perplexed. So, if the perplexity is high my model is very, very confused; if it is low my model is not so much confused. So, in this case, what you are seeing if I use a unigram model perplexity is 962 the model is very, very perplexed, but if I go for a bigram model, it is 170 the model is not so much perplexed now. Trigram model it becomes even better it is only 109, so that is what you are seeing unigram per perplexity of 962 means the model is as confused on the test data as if it had it had to choose uniformly and independently among 962 different possibilities for each word. And that you can also relate with your previous example, because every time it had it had to choose among 10 different possibilities for each digit because it is about giving a probability of 1 by 10 to each digit. So, now, you understood what the perplexity means.

(Refer Slide Time: 11:58)

The Shannon Visualization Method

Use the language model to generate word sequences

- Choose a random bigram ( $\langle s \rangle, w$ ) as per its probability
- Choose a random bigram ( $w, x$ ) as per its probability
- And so on until we choose  $\langle /s \rangle$

<s> I  
I want  
want to  
to eat  
eat Chinese  
Chinese food  
food </s>  
I want to eat Chinese food

Pawan Goyal (IIT Kharagpur) Evaluation of Language Models, Basic Smoothing Week 2: Lecture 5 8/16

So, once we have built a language model we can also use it for other tasks. So, one very interesting task is called Shannon visualization method that is can I use this language model to visualize or generate sentences. So, if you have read the original paper of Shannon on the mathematical information theory, so there he uses this method to generate various sequences of words. So, what is the idea? Suppose I have learnt a language model can I use that to generate various sequences of words or sentences. So, how do we actually apply this Shannon visualization method? So, suppose I have to generate a sentence, so what I will do, I will have to generate a sentence. So, I will first choose a random bigram that is starting the sentence as per the probability. So, what do I mean by that.

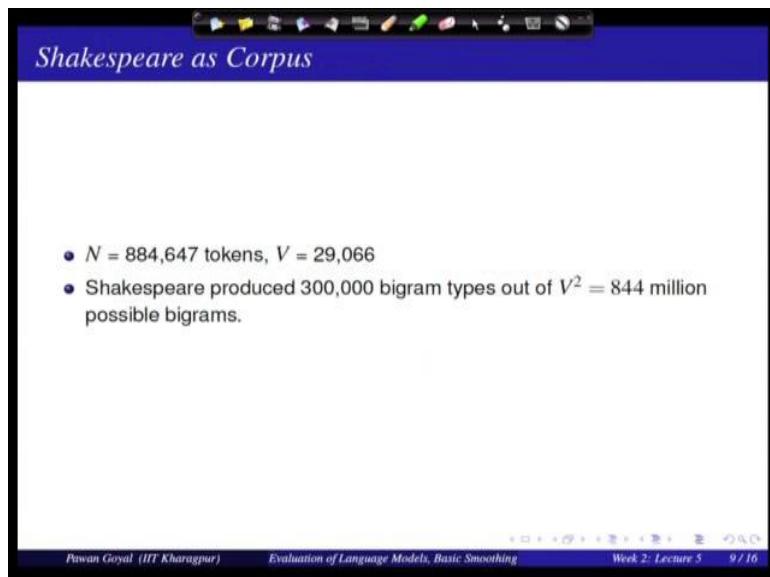
So, I am learning my bigrams. So, I have in with the start of the sentence whether the word  $w_1$  occurs with what probability. With the start of this sentence, word  $w_2$  occurs with some probability. So, this is a complete probability distribution that will add up to 1. Now, if I have to generate a sentence, I have to choose one among this possibilities and this will depend on the probability of that bigram. So, this you can think of as multinomial distribution and you are sampling one word from this multinomial distribution. So, suppose you pick picked up a particular word here. So, you start generating a sentence you start and  $w_2$ .

Now I have to choose the next word. So, that is the next step choose a random bigram as per its probability. So, what I will do now I will go to the distribution  $w$  to  $n$  different words  $w_1$ ,  $w_2$  and so on. And from this distribution again from this multinomial distribution I will

sample one sample one word suppose I find sum w 50. Again I will try to a sample word with its distribution w 50 and the next word. Now, the question is when do I stop when do I say that my sentence is complete, I will say my sentence is complete once I sample a word with the end of the sentence. At some point I sample w i and the next word is end of the sentence I say my sentence is closed. So, this is the whole idea of Shannon visualization method.

So, I do that until I choose end of the sentence and we will see one example from one corpus. So, from restaurant corpus that we discussed in the last lecture, suppose I have learned my bigram model and I want to generate a sentence. So, how will I do that I choose start of a sentence find out the first word, I find i, then I take the first word as I choose the next word how do I choose the words by sampling from the multinomial distributions. So, in this case, if we sample we might end up with getting the sentence I want to eat Chinese food and after food we get the end of the sentence. So, I want to eat Chinese food would be a sentence that is generated by this method.

(Refer Slide Time: 15:36)



Now, suppose we try to use this method over the Shakespeare's corpus. So, we have some number of tokens and the vocabulary size is 29,066. Just to give you an indication that bigrams are actually very, very sparse. So, if you take the vocabulary size of 29,000 something how many bigrams are possible. So, what do you mean by a bigram - two words together. So, I can take say  $V^2$  are the possible bigrams because every possible combination can occur, but so this gives me a number of 844 million possible bigrams, but in

the corpus how many bigrams were actually observed. So, we find there were only 3,00,000 bigrams that we have observed in the corpus. So, this is very, very sparse. So, now, suppose I build various language models from this corpus and try to generate various paragraphs and sentences. So, what do we actually observe, you will see what happens if I take unigram model, bigram model, trigram model and high order models.

(Refer Slide Time: 16:43)

**Unigram**

To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have  
Every enter now severally so, let  
Hill he late speaks; or! a more to leg less first you enter  
Are where exent and sighs have rise excellency took of.. Sleep knave we. near; vile like

**Bigram**

What means, sir. I confess she? then all sorts, he is trim, captain.  
Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.  
What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman?

**Trigram**

Sweet prince, Falstaff shall die. Harry of Monmouth's grave.  
This shall forbid it should be branded, if renown made it empty.  
Indeed the duke; and had a very good friend.  
Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.

**Quadrigram**

King Henry.What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in;  
Will you not tell me who I am?  
It cannot be but so.  
Indeed the short and the long. Marry, 'tis a noble Lepidus.

So, here you are seeing, if you take unigram model, you are getting some words that are probably very popular in Shakespeare's, but the sentence themselves are not making sense. So, you see the word like which here, it occurs independent of any other words and this becomes a complete sentence that will not probably something that you will see in the Shakespeare's corpus. So, now suppose you go to the bigram model now. So, you start making some sense. So, you have words sequences like what means, I confess and all sorts, they have good bigrams, but if you try to observe the sentence probably they are not making much sense.

Now, if you go to trigram model then again you have getting some more sense falstaff shall die and the shell forbid, it should be, should be brand, they have some nice sequences again and this is starts making a much more sense in terms of sentences. And if you go to quadrigram, then you are getting something that is resembling King Henry, What I shall go see the traitor Gloucester. So, this look like a valid sentences from the Shakespeare's corpus. So, that is the idea as you go to higher and higher order model, the kind of sentences you

generate will be something that actually resemble the corpus from which we are training this language model. So, this is my visualization method.

So, now we will try to go to the problem of a smoothing that we discussed in the last lecture. So, remember what the problem was in my language model suppose I am training for bigrams, so yes. So, just take the statistics that we saw from Shakespeare's corpus. There were 844 million possible bigrams out of which only 3,00,000 bigram actually occur. So, now if you assign the probability to each bigram very few will get a probability greater than 0, others will get a probability of 0. Now, suppose you are taking a test data and finding out how much it resembles Shakespeare's corpus. And whenever you see a bigram you are taking probability from the trained language model. Now, suppose a bigram occur that is not there in the Shakespeare's corpus, imagine the probability is 0.

Now, what happens to perplexity value remember this is simply the multiplication of all the different probabilities. So, this will becomes 0, so that will not be very, very helpful. So, you actually would like to give it some probability, so that this does not become zero that is why we will study topic of smoothing, how we can assign different probability values to get around this problem of 0s.

(Refer Slide Time: 19:26)

The slide has a title 'Problems with simple MLE estimate: zeros'. It compares a 'Training set' and 'Test Data'.

**Training set:**

- ... denied the allegations
- ... denied the reports
- ... denied the claims
- ... denied the request

**Test Data:**

- ... denied the offer
- ... denied the loan

**Zero probability n-grams:**

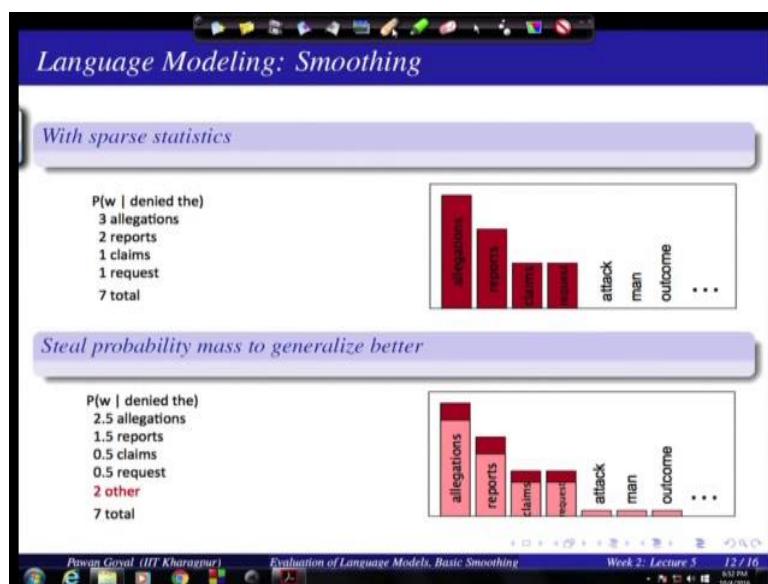
- $P(\text{offer} \mid \text{denied the}) = 0$
- The test set will be assigned a probability 0
- And the perplexity can't be computed

At the bottom, it includes the author's name 'Pawan Goyal (IIT Kharagpur)', the course name 'Evaluation of Language Models, Basic Smoothing', the week/lecture information 'Week 2: Lecture 5', and the slide number '11 / 16'.

So, let us take a simple example. Suppose I am learning my trigrams and my training data I have seen the following sentences. I have seen these many occurrences denied the allegations, denied the reports, denied the claims, denied the request and I am learning a

model of finding out the word after denied the. Now, suppose in my test data, I see these two occurrences, denied the offer and denied the loan. So, what would happen to the perplexity of my model? So, in the test data, I have seen these two occurrences that were not there in the training data. So, probability of offer given denied the will be 0, same with probability of loan given denied the. So, the test set will be assigned a probability of 0 and the perplexity cannot be defined, and that is what we were saying initially. So, how can I go around this problem? So that I can compute my perplexity even if there are certain bigrams or trigrams they did not occur in my training data. So, what is the idea of smoothing?

(Refer Slide Time: 20:38)



So, idea is suppose so this is what we were seeing. I am computing the trigram model for the word w after the occurrence denied the. And suppose in my data I see four different words right, I see allegations, reports, claims and request in total seven words. And I can assign the probability to each of these as 3 by 7, 2 by 7, 1 by 7, and 1 by 7 and that is what you are seeing in this plot 3 by 7, 2 by 7, 1 by 7, 1 by 7. So, these adds up to 1. Now, what about the probabilities for the other trigrams like denied the attack, denied the man, denied the outcome, these are also very, very feasible trigrams. So, here what will happen we will assign a probability of 0 to all of these yes. So, all these have a probability of 0. Now, what is the idea of smoothing? The idea of a smoothing is can I take some probability mass from each of these four words and assign that to the three words or any other words that I have not seen in my training data.

Can I steal some probability mass from these four words to assign some probability mass to the other words in my data? So, that is suppose here I have taken a probability mass of 0.5 each from the four words so that means, I get a probability mass of 2 by 7, and that mass I distribute among all the other words in my corpus. And this can be distributed in multiple different ways, we will see some possible ways in which we can distribute this stolen mass to the other words that we were dint see in my training data and how much mass has to be distributed that also we will see. So, there are different methods that do that. So, the basic idea is clear. So, how exactly we do that?

(Refer Slide Time: 23:09)

- Pretend as if we saw each word (N-gram) one more time than we actually did
- Just add one to all the counts!
- MLE estimate for bigram:  $P_{MLE}(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$
- Add-1 estimate:  $P_{Add-1}(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i) + 1}{c(w_{i-1}) + V}$

So, a simple method is called Laplace smoothing or add one estimation. So, what is the idea? So, pretend as if you have seen every n-gram one more time than we actually did in my training data, so that is suppose I saw it only once, so I will pretend as if I have seen it twice. So, what will happen to the n-grams that I have not seen at all in my training data? So I will pretend as if I have seen them once, so this is simple idea. So, we will just add one to their actually counts that we found from the training data, we will just add one to that. So, remember this maximum likelihood estimate for this bigram probability of  $w_i$  given  $w_{i-1}$ , we find it using number of times if we observe  $w_{i-1}$  followed by  $w_i$  divide by the number of times I observe  $w_{i-1}$  in my training data. So, this is my definition of MLE. So, how do I estimate language model or a bigram model using MLE?

Now, how do I use the Laplace's smoothing there? So, what did we say? We will pretend as if we have seen each n-gram one more time than we actually did. So, what we will do here. So, we will add 1 to the actual count. So, will do that for each possible bigram, but to ensure that the probability adds up to 1 I have to make some modifications to my denominator, so that they are normalized. So, what will I add to my denominator?

So, to get this answer, you can see how many different bigrams will be there for which I will be adding 1. So, how many different  $w_i$ 's will be there that will be number of words in my vocabulary; that means, I will add 1 capital V times. So, to normalize, I will also have to add a V here in the denominator and that is essentially the idea of my add one smoothing that I add a 1 to my actual n-gram count and add a V in my denominator to normalize it. So, this is my add one estimate number of counts plus 1 divide by the count of the unigram  $w_i$  minus 1 plus the vocabulary size, so that is the very, very simple smoothing technique that you can use in general. So, that does not require a lot of fancy estimates, you can just get your language model and easily apply this smoothing method.

$$c^*(w_{n-1}w_n)/c(w_n - 1) = (c(w_{n-1}w_n) + 1)/(c(w_{n-1}) + V)$$

(Refer Slide Time: 25:54)

Now, when we apply this add one smoothing method we can also talk about what is the effective bigram count. So, let me understand what do I mean by this effective bigram count. So, what I mean is, so you see here you have modified your actual counts or the probability, what is the idea? So, now, what is in effect what would have been the count in your actual

training data such that you have got the same would have got the same probability. So, what I am saying. So, this is your new probability, probability  $w_n$  given  $w_n - 1$  as per the add 1 smoothing, yes.

So, question here is what would have been my effective bigram count  $c \star w_n - 1$  such that if I do the MLE estimate PMLE,  $w_n$  given  $w_n - 1$ , I will get the same value as this. So, how do I get the effective bigram counts? So, this effective bigram count if I would divide by counter  $w_n - 1$ , I should have got this probability, so that is means I can compare  $c$  this probability with this probability and that will give me what is my effective bigram count. So, that is how I define my effective bigram count. So, we will see some example. So, if I apply this for my restaurant corpus, so what kind of effective bigrams do we observe.

(Refer Slide Time: 27:43)

*Comparing with bigrams: Restaurant corpus*

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16

So, we remember this is what this is the counts that we see in my bigram, this my restaurant corpus. So, we saw that in the last lecture, I want occurs very high number of time sense and so on. So, this is the actual bigram that we see. Now, suppose I apply add one smoothing, so you cannot apply to this data right exactly because is not the complete data this is just a small screenshot. So, now suppose you use add one smoothing and then if you find that what is my effective bigram counts. So, what kind of effective bigram count do you see? You see in this table certain counts are 0. So, one thing you would assume that after applying this smoothing technique, these will not be 0; this will be greater than 0. And whatever for having a high

value should have a smaller value because some mass would be stolen from there to give it to the words they did not occur and that is what exactly you will see when we apply this add one smoothing and then do the effective bigram counts. So, these are my effective bigram counts.

So, you see here whenever the word, I want the bigram I want occur 827 times initially the effective bigram count now is only 527. On the other hand, I too did not occur at all in my training data, but now it has effective bigram count of 0.64. Another observation you can make from here given the previous word for the next word if that occur 0 times the effective bigram count remains the same, so its 0.64 whenever the previous word is I for the words like to Chinese food and lunch, but this varies across different previous words. So, if you take the word like want, for wants this value becomes 0.3 time for to it becomes 0.63. So, this depends on the previous word. And that you can also see why because I am doing this plus v to the count of the unigram, yes and that will be different for different words that is why this value will be also different for different words.

(Refer Slide Time: 29:48)

*More general formulations: Add-k*

$$P_{Add-k}(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i) + k}{c(w_{i-1}) + kV}$$

$$P_{Add-k}(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i) + m(\frac{1}{V})}{c(w_{i-1}) + m}$$

Unigram prior smoothing:

$$P_{UnigramPrior}(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i) + mP(w_i)}{c(w_{i-1}) + m}$$

w<sub>i</sub> | w<sub>i-1</sub>      w<sub>i</sub> going to w<sub>2</sub> Smoothing      w<sub>i-1</sub>

Pawan Goyal (IIT Khararapur)      Evaluation of Language Models, Basic Smoothing      Week 2: Lecture 5      16 / 16  
7:04 PM      16/4/2018

So, in general, so here we talked about the add one estimation, but in general there are some simple variants of this also. So, one simple variant is called add-k estimation. So, we are adding one to each bigram. So, why want one, why cannot we do some general thing like k, k can be 0.5 or more than 1 depending on how big your data is. So, this is called add-k estimation.

So, here what is the idea,

$$P_{Add-k}(w_i|w_{i-1}) = (c(w_{i-1}, w_i) + k)/(c(w_{i-1}) + kV)$$

we add-k to each count and accordingly we will add-kV to my denominator, yes. So, this is actually the same as add one estimation if I take k is equal to 1 .

Now, I can also make some variant here. So, suppose I say KV is equal to m. So, now, I can write it as this plus m, and this plus m by V, yes, this is effective the same, now m is k times v. So, this is another variant. And this also gives me an idea on how I can improve this basic smoothing method. So, here what I am doing, I am adding m by V to each word effectively I am doing m times 1 by V to all the V words in my vocabulary. And if we add up for all the words in a vocabulary m by V, you get an m. So, now, can we do something better there the idea is instead of adding m times a uniform one by V to each word can you add m times the unigram probability for the word. And you say this will add up to one for all the words you will effectively end up getting the same values in numerator and denominator when you normalize the probability, but this might be a better estimate, why is that.

So, let us first see all these variations. So, this is when I replace kV by m that is what I get. So, here in place of 1 by V, I can also replace the probability of the word that will be a different smoothing, but what we are saying this might be a better way of doing a smoothing this called unigram prior smoothing. So, let us just discuss this point why this might be a better way of doing a smoothing. So, remember what we are trying to do here, we are trying to find a probability or different words that do not occur in my training data  $w_i$  given  $w_{i-1}$ . Take a word  $w_i$  that is very, very common. So, like I have a word government that is very, very common and I might have some other word like may be something like smoothing that may be other word  $w_1, w_2$ .

What are you doing in your add one smoothing or add-k smoothing? You are just studying k or one and dividing by the count of the previous word plus kV and this will be same for both the words, yes. So, what is the idea of unigram prior smoothing? Idea is that if I know that this word is more common in my corpus then this word probably I can say that the probability of this word occurring of a  $w_{i-1}$  will also be higher than the probability of this word occurring after  $w_{i-1}$ , and that is I am trying to exploit. This data I have from my corpus and I am trying to exploit that for doing my smoothing. So, this is called unigram prior smoothing. So, in general so there are many other ways of doing smoothing also.

So, we discussed add one smoothing and the institution behind unigram prior smoothing, but there are other advance models of smoothing also that we will see in the next lecture, so in week 3.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 11**  
**Tutorial**

Hello everyone. Welcome to the first tutorial session for the Natural Language Processing course. In this tutorial session we will be seeing how to set up the programming environment for the course and how to use it to solve your assignment problems. So, we will be using Python as our default programming language and we will be using Jupyter as our default programming environment.

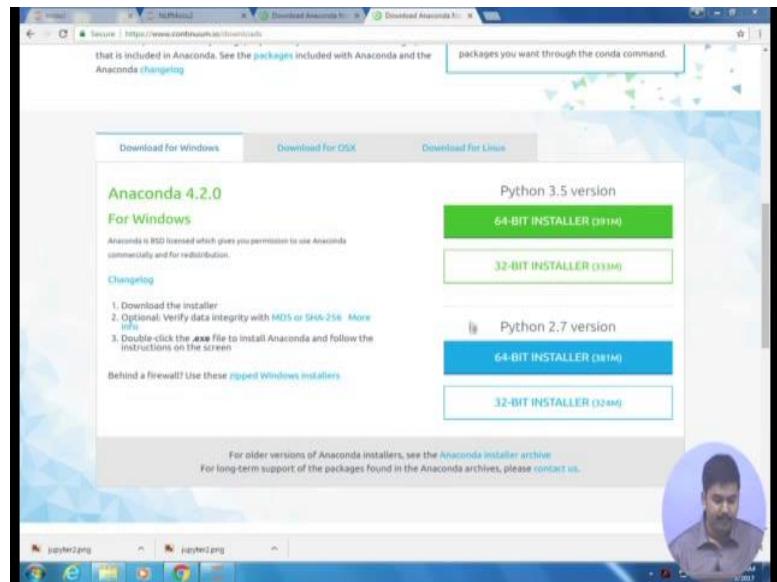
So, if you are already comfortable with a particular programming language or an environment it is perfectly fine that you use one of those, but if you feel that you will be requiring our assistance it is recommended that you use Python and Jupyter.

(Refer Slide Time: 01:16)



So, let us see how to install a programming environment. So, what I will suggest is that you can go to Google and type Anaconda Python. So, Anaconda is a package that gives you the programming language, programming environment and all other necessary packages required to run your course. So, you can see that the first link here which is the Anaconda package leads you to the download page. So, the (Refer Time: 01:46) we get here we can go to the menu and see the download link is up here.

(Refer Slide Time: 01:58)



For installation we can look into the packages here. As we can see the package is available for different operating systems and we recommend that you go with the Python 2.7 version.

So, depending on your OS you can install the 60 bit installer or the 323 bit installer. So, if you are working on OSX or Linux where installers are available here we recommend that you install that (Refer Time: 02:30) version and go to the particular director where the package is downloaded and run your dash command to execute this particular package. The command line installer will take you through and it show you how to install the package.

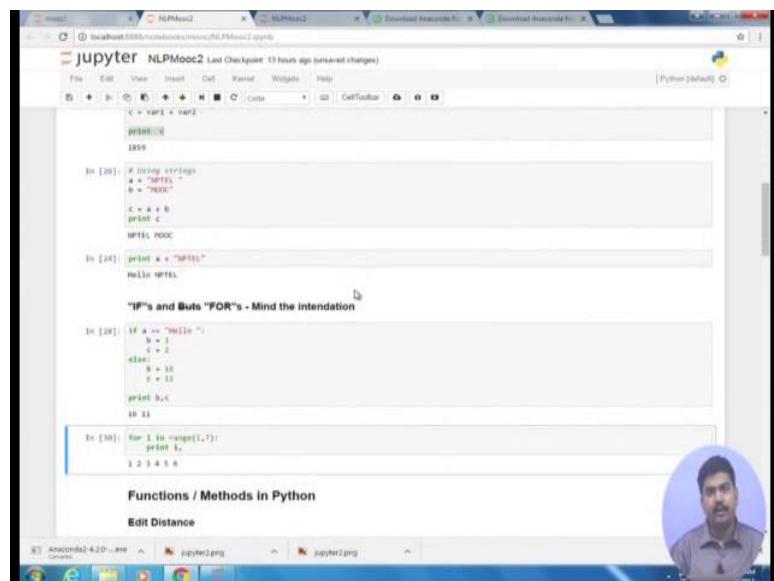
For the windows the graphical installer is available. And you can download it by clicking on this particular link. As I have already have the install version available with me I will be showing you a step by step process of how to install this set up. So, if you have space constraint you can cry the minimal package called as Miniconda otherwise it is advisable that you go with the current installation which is Anaconda.

So, here is the installation package and you can go to a step by step process. And yes, it ask for two separate ways of installation if you do not have admin privileges in the particular system you are installing you should be selecting this one just for your particular use. And you can define why you are package should be install and make sure you have at least 2 GB of free space available in your system. Since just that I have

already a package install. So, I will be just showing you how to install it to on a separate folder and I just you have to make sure this are ticked and click on the install button.

So, the install will take somewhere around 15 to 20 minutes to complete. And once its complete you will be able to see the Anaconda package in your programs and once you click here we can see different packages installed where you should be selecting Jupyter notebook. So, Jupyter notebook is a programming environment that runs from your browser. So, it has a server running at the back. And I have an instance of Jupyter already running in my browser.

(Refer Slide Time: 05:05)



So, it goes with by default goes to your document folder and within the document folder it will show you where exactly your files are residing. So, as part of the assignment we will be providing you some help of files of which this is the particular programming file that you should be looking for. So, this is your programming environment.

So, notebook is not just a programming environment it helps you keeps lot of things that you learn, it helps you document the stuff that you basically go through so you can refer it back. So, I will be using Python as my language inside Jupyter. And Python also tends to be my favorite calculator. For example, if you want to calculate just some values I can just see how multiplication of two values this may an answer. So, 987 into 543 will give me this particular answer. Now, if I want to stop using my variables I can give the values to the variables there is no type declaration as required, we can just give a

variables name so maybe I can give variable 1 and this will be variable 2 and c equal to variables 1 plus variable 2; and then I print the value for c.

Now, assuming since we will be we are studying about natural language processing we will be taking a lot of things and lot of textual data to handle. We how to stop text strings. So, if you have variables you can just assume text strings enclosed with in quotes. So, here I have typed ‘hello’ and world and I do the plus operation to both the variables let see what happens. So, the particular variables strings hello world together which means it concatenation in a both the variable. So, this automatic identification whether the sum needs to be done or concatenation needs to be done this handled by the Python interpreter.

Suppose if I change and I say ‘hello mook’ let us say how with becomes, yes; so the output given here basically shows ‘hello mook’. Similarly, a variable initialized at a particular cell can be reused at another cell as it resides in your memory. For example, I reused the ‘hello’ which was already defined in the particular cell here and I use hello in NPTEL let see how it works.

An important thing that you have to care of is indentation. So, when using conditional statements like the ‘if’ or the loops like for loop you have to make sure that the block indentation is maintained. So, what I want to check is that if ‘a’ is equal to hello. So, I check for whether the variable ‘a’ contains the variable hello then the variables are given the values 1 and 2 otherwise the variables b and c are given the values 10 and 11. Let see which value gets print.

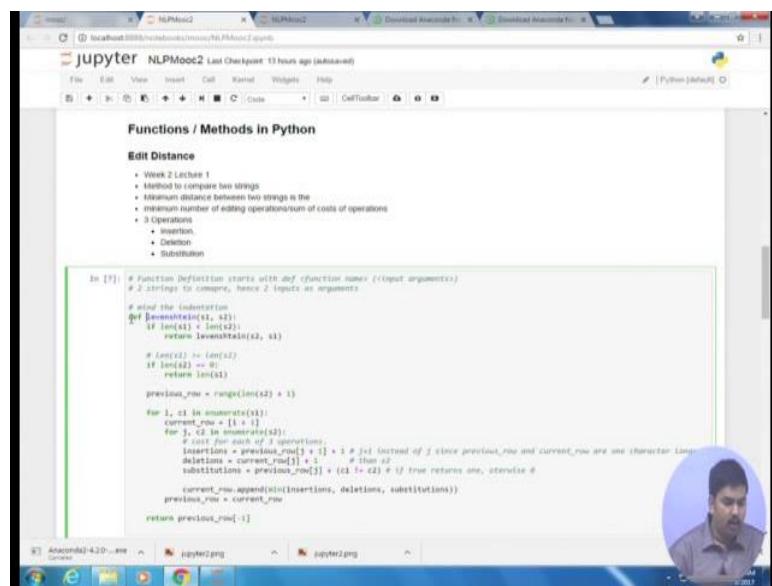
Since the variable a has the value hello it prints 1 and 2. Suppose I change it to NPTEL and execute it and now I run here. So, here I get the values 10 and 11. So, if you are still confused how to run the code after typing the code in the particular cell we can either use the shortcut control and enter or you may go to this button and this will execute the code. So, let us change the variables values here and see if what happens if they enter code. Now b gets the values 60 and that gets printed here.

Now, suppose I am using a loop to print all the values between 1 and 5 that is 1 to 4 how do I do. So, I defined keyword 4, I define a variable and I define that the variable will be within a range from 1 to 5. If we can make it 7 or any value of your choice and let us see.

So, it prints all the values from 1 to 6. If I do not want to print it line by line I can just use a comma here and it gives me the values written in the same line.

So, this is some basic syntax that will let you coding Python and we recommend that you go through some of the basic tutorials regarding Python and some of the basic data structures that Python has so that it will help in making coding experience much better.

(Refer Slide Time: 10:57)



The screenshot shows a Jupyter Notebook interface with a title bar "jupyter NLPMooc2 Last Checkpoint 13 hours ago (auto saved)". The main area displays a code cell titled "Edit Distance". The code implements the Levenshtein distance algorithm:

```
# Function definition starts with def function name (<input arguments>)
# 2 strings to compare, hence 2 inputs as arguments
def levenshtein(s1, s2):
    if len(s1) > len(s2):
        return levenshtein(s2, s1)

    if len(s2) == 0:
        return len(s1)

    previous_row = range(len(s2) + 1)
    for i, c1 in enumerate(s1):
        current_row = [i + 1]
        for j, c2 in enumerate(s2):
            insertions = previous_row[j + 1] + 1 # i+1 instead of j since previous_row and current_row are one character longer
            deletions = current_row[j] + 1
            substitutions = previous_row[j] + (c1 != c2) # if true returns one, otherwise 0
            current_row.append(min(insertions, deletions, substitutions))
        previous_row = current_row
    return previous_row[-1]
```

In order to provide that we will be providing you some links and some resources that you can go through and get fresh up with the particular code. So, as part of your assignment you are supposed to solve questions that involved edit distance that involves calculating the likelihood using bigrams and unigrams and I will be showing you how to handle this.

As from the week two lecture one you might be aware of the concept called as Edit Distance. So, edit distance is a particular function that helps you compare two strings and find the similarity between them. So, the edit distance gives you the minimum distance between two strings and the minimum distance can be calculated in terms of the number of operations that required to change one string from to another. And this is defined in terms of three operations insertion, deletion and substitution.

So, we have provided a function called Levenshtein that helps you execute or calculate the edit distance. So, this is how functions are methods are used in Python we use the keyword def followed by the name of the function. And since I have two strings to come

I give the strings as my function arguments that, so that when you call the function you have to give which are the two strings to be given.

Now, these are the three operations; the insertion, deletion, substitution that you will be using to compare both the strings. So, we need not worry about what exactly is happening inside you can think of it as a black box and you can just execute it. But if you are curious, you can obviously work around with it hack around with it and see how the output changes.

So, assuming I already have the code setup and I already executed this particular cell so that the function goes to gets invoke. Now I call this function Levenshtein and I give two strings which is h e l l p and h e l l o and I want to see what will be the difference between the strings or what will be the edit operation edit distance. It turns out that the edit distance is 1 which is being written by this particular variable here.

(Refer Slide Time: 13:40)

The screenshot shows a Jupyter Notebook interface with two code cells and their outputs.

```

In [34]: def Levenshtein(str1, str2):
    previous_rnd = [0]
    for i in range(len(str1)):
        insertions = previous_rnd[i] + 1 # If current_rnd[i] and current_rnd[i+1] are one character longer
        deletions = current_rnd[i] + 1 # then add
        substitutions = previous_rnd[i] + (1 if str1[i] != str2[i] else 0) # If true return one, otherwise 0
        current_rnd.append(min(insertions, deletions, substitutions))
    previous_rnd = current_rnd
    return previous_rnd[-1]

In [34]: Levenshtein('bleal', 'vimele')
Out[34]: 1

```

**Jaro-Winkler Distance**

The Jaro-Winkler distance is a measure of similarity between two strings. The Jaro-Winkler similarity is given by  $\lambda - \text{Jaro-Winkler Distance}$ . The Jaro-Winkler distance metric is designed and best suited for short strings such as person names. The similarity score is normalized such that 0 equates to no similarity and 1 is an exact match.

$$d_j = \begin{cases} 0 & \text{if } m = 0 \\ \frac{1}{3} \left( \frac{m}{n} + \frac{m}{k} + \frac{m - t}{2k} \right) & \text{otherwise} \end{cases}$$

Where:

- $n$  and  $k$  are the strings
- $m$  is the number of matching characters (see below)
- $t$  is half the number of transpositions (see below)

```

In [36]: from editdistance import jaro_winkler
In [36]: jaro_winkler('bleal', 'vimele')
Out[36]: 0.9999999999999999

```

**Handling Files**

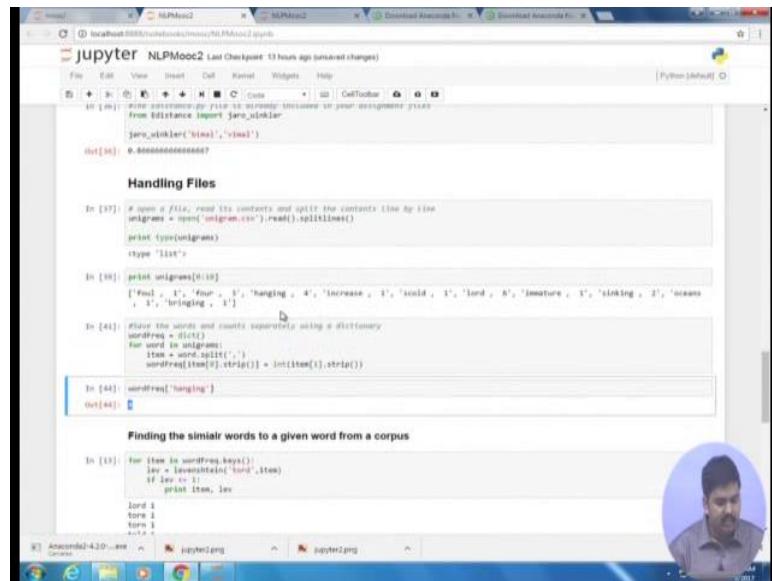
Suppose I want to give; yes it also gives the same values. So, if I have a new string the value changes. As part of your assignment you have one question where you are supposed to solve the problems using Jaro-Winkler distance. Jaro-Winkler distance is a variant of edit distance that is used to combine strings, and we have provided the explanation of how to calculate the Jaro-Winkler distance between two strings. And you can read about more in the Wikipedia link provided that.

For your convenience I have we have already implemented the Jaro-Winkler as a function and we have stored it in the packages edit distance. So, we can call the Jaro-Winkler function and you can again assume it to be black box just to solve your assignment. So, in Jaro-Winkler as you can see the value ranges between 0 to 1. And if two strings are exactly the same they will provide you the value of 1 and if they are complete different they will provide a value of 0. So, if I give the same string I get a value of 1 as my Jaro-Winkler distance.

So, this is very useful when it comes to combine strings such as person names because there are each variations in the person name that we use. For example, it is very common that the word Vimal might be spelled as Bimal depending on your demography. So, suppose we apply this function and let see how the string gets the value. So, the similarity between Bimal and Vimal turns out to be 0.867.

Now let us see given a data set or a corpus of words and you want to find the most similar words of a given string from that how do we do that.

(Refer Slide Time: 15:37)



The screenshot shows a Jupyter Notebook window with several code cells and a video feed of a speaker in the bottom right corner.

```

In [39]: # write a file, read the contents and split the contents line by line
unigrams = open('unigram.csv').read().splitlines()
print type(unigrams)
type('list')

In [40]: print unigrams[0:10]
['Aard', '1', 'Afar', '3', 'Hanging', '4', 'Increase', '1', 'could', '4', 'Lord', '8', 'Immature', '1', 'sinking', '2', 'oceans',
 '1', '2', 'Bringing', '1']

In [41]: #use the words and counts separately using a dictionary
wordfreq = {}
for line in unigrams:
    item = line.split(',')
    wordfreq[item[0].strip()] = int(item[1].strip())

In [44]: wordfreq['hanging']
Out[44]: 4

```

**Finding the similar words to a given word from a corpus**

```

In [13]: for item in wordfreq.items():
    lex = levenshtein('Lord',item)
    if lex <= 1:
        print item, lex

Lord 1
Lord 1
Lord 1
Lord 1

```

So, you will be provided with a file called unigram dot csv, you need to load that file and you have to. So, every line contains a single word or every unique word and the frequency with which it appears in a particular purpose.

So, I read the file to the variable unigrams; and unigrams will be stored as a list there is a default data structure used in Python you can ignore about list from the Python manuals. So, for convenience you can think of it as an array where if I want to access the first element I just give the index of the first entry which is 0 and I print that well. So, the entry gives me foul for which has a frequency of 1. Suppose I want to say the first ten entries in the file I will use the colon followed by number 10, so it will give me all the numbers from 0 to 9 and that gives me the first 10 entries which is foul which appears 1 time foul which appears 3 times and immature appears 1 times bringing appears 1 time.

Now, I will be using a different data structure called as dictionary that helps me separate the count and the word from one another. So, dictionary as the name suggest we have a key where the key is your word and when I provide the key to the dictionary it will provide me the count with which it is appearing. So, here is the function that helps me go and it stores the value inside the dictionary called as wordfreq.

So, once I execute it. So, a dictionary has two entities; one it is called as the keys and it has it made as a list of the keys. So, if you type wordfreq dot keys it will provide you all the keys that it has. Now if I want to know how many times thunder is appearing, what I would do is I would see the count of thunder by just giving the key as thunder to the variable wordfreq. Now it appears only once let us see how many times hanging is appearing and that is validate with that it is coming out to be correct. It is coming out to be correct as its 4 here and we already have checked it was 4.

Now, since you know how to open a file how to store it in a dictionary and you already also know how to compare two strings, let us see we can whether we can find the most similar word to a given string or a given word.

(Refer Slide Time: 18:46)

The screenshot shows a Jupyter Notebook interface with two code cells. The first cell, In [37], contains code to read a CSV file and split its contents into a list of unigrams:

```
# open a file, read its contents and split the contents line by line
unigrams = open('unigrams.csv').read().splitlines()

print type(unigrams)
type('list')
```

The second cell, In [38], prints the first few items from the unigrams list:

```
In [38]: print unigrams[0:10]
['Paul', '1', 'Four', '3', 'hanging', '4', 'increase', '1', 'scold', '1', 'Lord', '6', 'immature', '1', 'sinking', '2', 'oceans', '1', 'tore', '1', 'Stringing', '1']
```

The third cell, In [41], defines a function to count words in a dictionary:

```
In [41]: #use the words and counts separately using a dictionary
wordFreq = {}
for item in unigrams:
    word = item.split(',')
    wordFreq[word[0].strip()] = int(item[1].strip())
```

The fourth cell, In [44], uses the wordFreq dictionary to find the frequency of the word 'hanging':

```
In [44]: wordFreq['hanging']
Out[44]: 4
```

The fifth cell, In [48], finds words similar to 'tord' using Levenshtein distance:

```
In [48]: for item in wordFreq.keys():
    lev = levenshtein('tord',item)
    if lev <= 2:
        print item, lev
```

The output shows five words with a Levenshtein distance of 1 or less:

```
Lord 1
tore 1
torn 2
old 2
tore 1
tore 1
tore 1
word 2
```

So, what I do is that I iterate through all the possible words in the dictionary and then I find the Levenshtein distance between the string that I want to come here which happens to be t o r d. It is a spelling mistake. So, I have to which word be replaced with t o r d.

And if you see here I take each word from the keys of the dictionary and assign it to the variable item one at a time and I compare tord with item. And I am going to print all the words that has a Levenshtein distance which is less than or equal to 1. So, if you fine tord well and good otherwise let us see what are the other words.

Once it turns out there are five words which are similar to t o r d; which is lord tore torn told or word. Let us see all the words which have a Levenshtein distance of less than or equal to 2 when it turns out there are much more entries which have a Levenshtein distance of two with words.

(Refer Slide Time: 20:12)

The screenshot shows a Jupyter Notebook interface with several code cells and their outputs. The code is as follows:

```
In [14]: bigram = open('bigrams.csv').read().splitlines()
wordbigram = dict()
for user in bigram:
    item = user.split(',')
    wordbigram[item[0].strip()] = int(item[1].strip())
    
Calculating bigram likelihoods from a corpus
In [17]: print(wordbigram['come back'], wordfreq['come'])
10 146
In [18]: bigramprob = wordbigram['come back']/wordfreq['come']
KeyError: 'come'
0
In [19]: bigramprob = wordbigram['come back']/(wordfreq['come']+1)
print(bigramprob)
0.00002315068489
In [20]: #### Add one smoothing
add_one = (wordbigram['come back']+1.0)/(wordfreq['come']+1.0)+len(wordfreq.keys())
print(add_one)
0.00012385531909 0.022
In [21]: #### Installing and using functionalities in an external package
#using a package
import gensim
#### not working? try install
ImportError: Traceback (most recent call last):
  File "C:\Python34\lib\site-packages\gensim\__init__.py", line 10, in <module>
    from . import _gensim

```

Now what will be doing either you are given with another file called the bigrams dot csv, and it contains all the bigrams with the frequency with which the bigrams are appearing.

So, this is very straight forward, just like what we have done previously for unigram dot csv we are going to use the same set of code if you see here. But here I have just written it in 1 lag. So, I have the bigrams file which I have read and I have spitted into a list then I define the dictionary called wordbigram and I store every bigram as my key and the count as the value. So, bigrams and unigrams as tord as now two different dictionary; one is in wordfreq and other is in wordbigram.

Now, suppose I want to calculate the bigram likelihood. As we know the formula is the count with which the bigram occurs by the count with which the first word in the bigram occurs. So, if I want to calculate the bigram likelihood of comeback I want in other frequency with which the comeback is occurring in a particular corpus and the frequency with which come is occurring. And it turns out that the value is occurring 10 times and come is occur 140 times. So, if I want to calculate the probability of come back and come I would do wordbigram come back divided by word frequency of come which will give me 10 by 146.

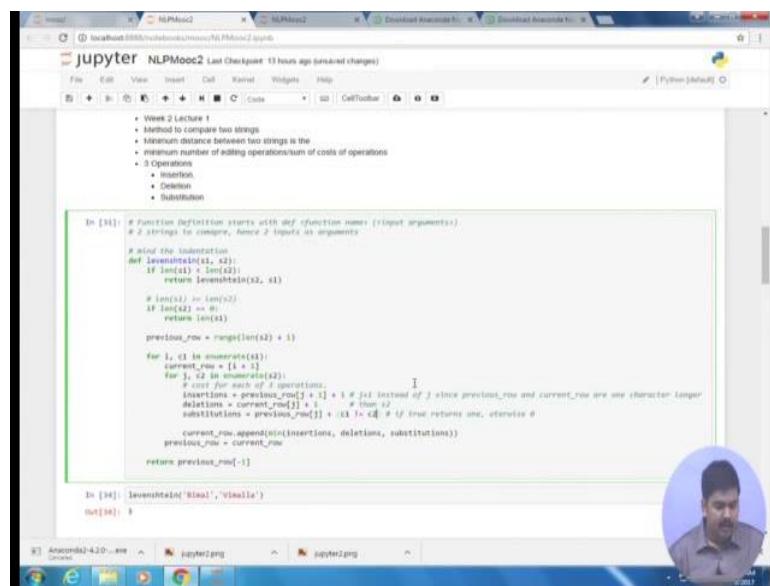
It turns out that the value gives coming out to be 0 so there is certainly some problem. The problems is that since both come the frequencies are individual values the system assumes the answer also should coming an integer, but it turns out that the value is

coming out to be a floating point number. So, to handle up there are different ways of explicit type casting what I am doing here is just multiply the value with a 1.0 so that before the division occurs one of the value turns out to be a float. After the operation is done the value turns out to be 0.068.

Now suppose we want to calculate the likelihood of the same bigram, but using add one smoothing as we know. Add one smoothing basically adds a 1 to the numerator for the bigram likelihood and it also adds to the vocabulary size or the number of unique words in a corpus. To get the number of unique words in a corpus we can just take all the keys in the dictionary and we can find the count of that and the number of keys. And it turns out that the function length performs the same. So, I have used the length function to find all the length of the keys. And with that I print the, add one likelihood after add one smoothing. And we can also see how many unique words are there in the corpus.

So, this is pretty much for what you can do with your assignment 2. And there will be one question where you are supposed to modify the course in word for the particular operation.

(Refer Slide Time: 24:10)



```

In [31]: # Function definition starts with def function name (input arguments)
# 2 strings to compare, hence 2 inputs as arguments
# and the operation
def levenshtein(s1, s2):
    if len(s1) > len(s2):
        return levenshtein(s2, s1)

    # len(s1) <= len(s2)
    if len(s1) == 0:
        return len(s2)

    previous_row = range(len(s2) + 1)
    for i, c1 in enumerate(s1):
        current_row = [i + 1]
        for j, c2 in enumerate(s2):
            # cost for each of 3 operations
            insertions = previous_row[j + 1] + 1 # j+1 instead of j since previous_row and current_row are one character longer
            deletions = previous_row[j] + 1 # i < c1
            substitutions = previous_row[j] + (c1 != c2) # If true returns one, otherwise 0
            current_row.append(min(insertions, deletions, substitutions))
        previous_row = current_row

    return previous_row[-1]

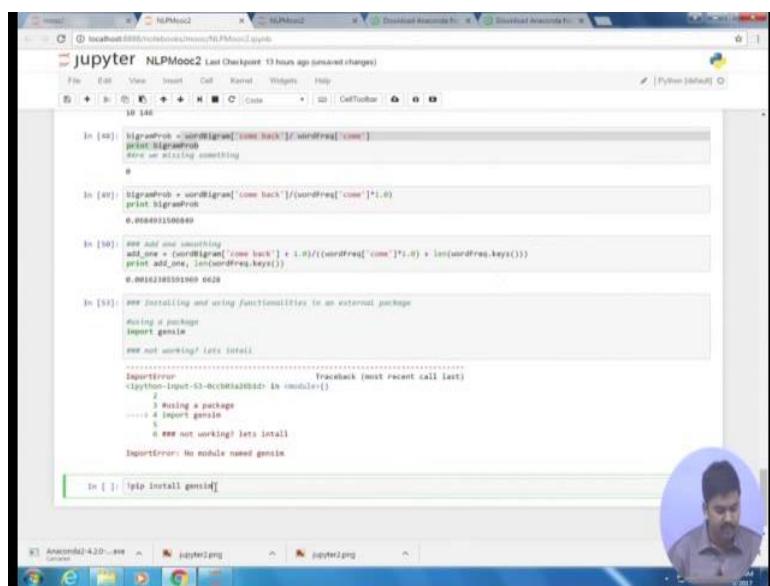
In [34]: levenshtein('Blah', 'Vimelle')
Out[34]: 3

```

So, in that particular question you are supposed to change the value for substitution. As insertion and deletion involve at cost of 1, but substitution involves a cost of 2; in order to do that I am going to modify the value.

So, what this particular function does is that it compares whether two characters at a position are same. So, if they were not same it returns a 1 because this is a logical operation otherwise it returns a 0. So, previously the cost was 1, so it was left (Refer Time: 24:54). Now since the cost is two and just modifying it to be two here. And let us see how much does it cost and how Bimal and Vimal compare with this (Refer Time: 25:05) operation. And it turns out that the value becomes 2. This is generally done when to want to penalize substitutions and you want to check more into the insertions and deletions operations.

(Refer Slide Time: 25:26)



The screenshot shows a Jupyter Notebook interface with several code cells and an Anaconda Navigator window in the background.

```

In [48]: bigramprob = wordBigram['cone back']/(wordFreq['cone'])
        # we are missing something
        0

In [49]: bigramprob + wordBigram['cone back']/(wordFreq['cone'])*1.0
        print(bigramprob)
        0.000000000000000949

In [50]: new_and_one_smoothing
add_new = (wordBigram['cone back'] + 1.0)/(wordFreq['cone']*1.0) + len(wordFreq.keys())
print(add_new, len(wordFreq.keys()))
        0.00162380591960 6628

In [51]: %% installing and using functionalities in an external package
        # Using a package
        import gensim
        #%% not working? lets install
        ImportError: Traceback (most recent call last):
        /c/python35/python35/lib/python3.5/importlib/_bootstrap.py:219: In <module>
            _check_type(module_name, module)
        ...
        3 # Using a package
        .... 4 import gensim
        0 #%% not working? lets install
        ImportError: No module named gensim

```

In [52]: !pip install gensim

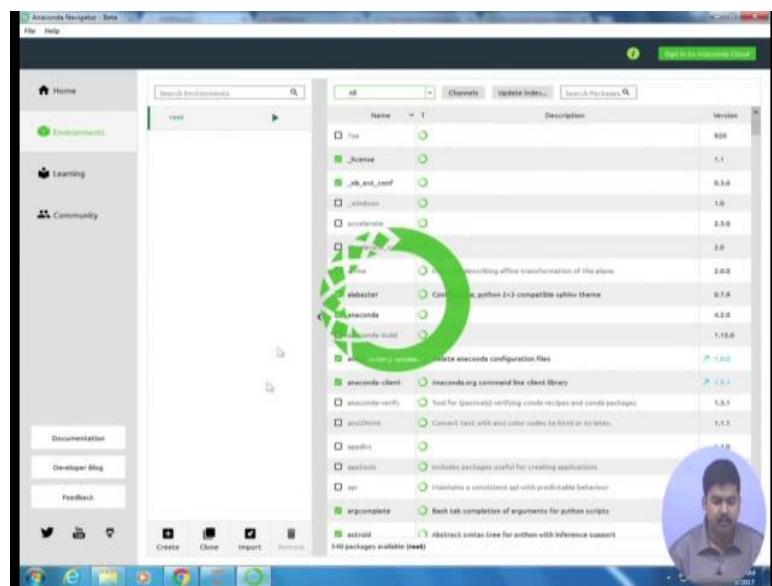
And for some aspects you might require some external packages to be installed and I will be showing you how to install; an external packages. Suppose I know there exist a package gensim and I assume that it is installed in this system. I will invoke that external package using import gensim. But, it turns out that the package is not installed. So, Anaconda comes with a default package manager which you can select using here Anaconda navigator. So, Anaconda navigator provides you all the different packages and it provides a repository way of different packages that you can install.

Alternatively, you can create a new cell and you can install a package within the notebook itself. What you have to do is that you have to mind this exclamation mark so you have type pip install gensim and this will also help you install the particular package.

But you can have in trouble with that you can by default go to the Anaconda navigator and it will help you install the (Refer Time: 27:59).

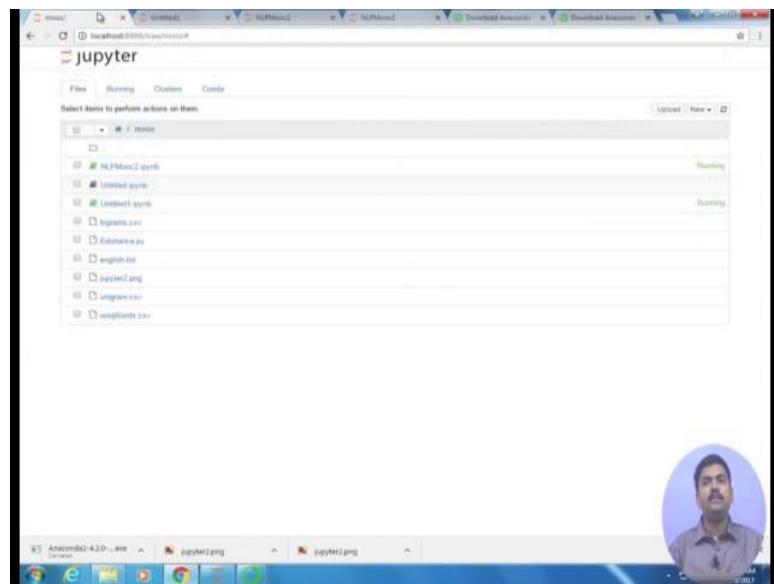
So, I will just show you how to install this same and that will be the end of the tutorial session.

(Refer Slide Time: 27:27)



So, here you can click on to the environment tab and you can see for all packages are reliable in the repository of Anaconda and you can just type gensim and you can click here you can say electric and you can apply it to install. As I was suggested you may alternatively use this command to install.

(Refer Slide Time: 27:56)



If you want to create a new notebook you should click on the new and you should select one of this. So, this will give me a new notebook where you can have a fresh notebook, where you can try out code by yourself. So, this is pretty much.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 12**  
**Language Modeling: Advanced Smoothing Models**

So, welcome back for the third week of this course. So, in the last week, where we ended we were discussing about language modeling, we talked about the basis of language modeling then we talked about what how to evaluate a language model and then we came to the topic of smoothing. So, what was the simple idea we discussed, what is my advanced smoothing where I add one to each individual count, and accordingly I do some normalization in my denominator. And we added by saying there are certain simple variance possible like at K, I can also do the unigram prior smoothing by instead of adding uniform weight to each word, at a weight that is depending on the probability of that word unigram probability for that word. And that we were arguing that might work better than simply adding uniform count to each of the bigrams.

There we had certain parameters like what will be a best way of choosing K and M in different models. So, simple answer would be, so there is no unique value that you can choose. Suppose you have some held out data, so we can try to find out for which value of K and M, you are getting the best publicity for that held out data and that can be one possible way of choosing the values of K and M. So, now today, we will talk we will go beyond the advanced smoothing and talk about some other advanced smoothing methods.

(Refer Slide Time: 01:52)

So, we will start with two different smoothing methods called Good-Turing smoothing and Kneser-Ney smoothing. So, what is the intuition behind using a Good-Turing smoothing? So, intuition is essentially I want to find out what should be the probability for the words that I have not seen in my training data. So, what Good-Turing smoothing says; so why do not you use the estimate of the things you have seen once in your training data you estimate about the things we have not seen. Similarly, whatever you have seen twice use that estimate that things you will see once and like that.

So, this just thrice to adjust the probability mass, so that the things that was seen twice are used to find the probability for the things that was seen once things that was seen once are used to estimate the probability for the things that were not seen at all. So, the idea is that you we have to use the count of things we have seen once to find the count of things we have never seen.

(Refer Slide Time: 03:02)

Word	Frequency	$N_c$
I	3	$N_1 = 4$
am	2	$N_2 = 1$
here	1	$N_3 = 1$
who	1	
would	1	
like	1	

So, for that let us first define what is the frequency of frequency. We used this one of the earlier lectures also. So, we take this three sentences, I am here, who am, I would like. From these sentences, I am trying to construct the frequency of frequency, so that is how many words occur ones, how many word occurs twice and so on. So, if you see here, the word I occurs thrice, am occurs twice, and the other four words occurs once. So, what will my frequency of frequency? I will find out how many of words occurs ones with frequency one. So, that is my  $N_1$ . So, there are four words here that occur only once. So,  $N_1$  is 4. Now, you will see how many words occur twice, so how many words have a frequency of 2, so that will I will have 1 here  $N_2$  is 1. So, this is my frequency of frequency. So, four words occur once, one word occurs twice, one word occurs thrice.

(Refer Slide Time: 04:07)

Now, how do I use that in Good-Turing estimate? So, what is the idea? So, this we discussed earlier. So, I want to reallocate the probability mass of n-grams that occur  $r + 1$  times in the training data to the n-grams that occur  $r$  times, especially you want to see the n-grams that occur once to find the probability mass for the n-grams that occur 0 times, but you that you do that in general. So, the n-grams that occur  $r + 1$  times used that for  $r$  times.

So, now in particular, reallocate the probability mass for n-grams that were seen once for those that were never seen. So, now, we should see formally what will be the effective probability mass N count that we will get by this Good-Turing estimate. So, let us see suppose I want to find out four words that occurred  $c$  times, words that occurs  $c$  times what will be the effective adjusted count after applying Good-Turing estimation, now how we defined that? So, what would happen? In my training data, I will have words that occur zero times words that occur one time let it be  $N_0$ ,  $N_1$  words that occur twice  $N_2$  and so on. So,  $N_2$  numbers that occur twice, similarly now there are  $N_c$  number of words that occurs  $c$  times  $N_{c+1}$  words that occurs  $c + 1$  times.

$$c^* = ((c+1)N_{c+1})/N_c$$

Now, I want to find I want to use Good-Turing to estimate the count for these words that occur  $c$  times. So, what it be say we said we will use this probability mass and give it to these words. Now how to we distribute. So, what is the probability mass associated with the words that occurs  $c + 1$  time. So, now suppose there are  $N$  words or  $n$ th bigrams in total. So, now

what is the probability mass for the bigrams that occurred or n-gram we can say in general that occurred c plus 1 times, you see how many n-grams are there are N c plus 1 different n-gram. So, what will be the probability mass for them it will be N c plus 1 times c plus 1 divided by the total number of n-grams that are N that is a probability mass, yes, and this I am distributing to the words that occur c times. Now, how many words are there, there are N c words?

So, what is the probability that each of them will get divided by N c? Yes, so this is the probability mass for that each of them will get. Now, what will be the effective count remember how we define the effective count c star such that if I divided by n merely this will give me the same probability, so that will tell me c star is N c plus 1 times c plus 1 divided by N c and that is what we have written here. So, N c in general is the n-grams that seen exactly c terms in the corpus. So, by using this definition I can find out what is effective count for any word that is occur N c number of times in the corpus, so that is what we have seen.

(Refer Slide Time: 08:10)

*Good Turing Estimation*

*Good Turing Smoothing*

$$P_{GT}^*(\text{things with frequency } c) = \frac{c^*}{N}$$

$$c^* = \frac{(c+1)N_{c+1}}{N_c}$$

*What if c = 0*

$$P_{GT}^*(\text{things with frequency } c) = \frac{N_1}{N} \text{ where } N \text{ denotes the total number of bigrams that actually occur in training}$$

Pawan Goyal (IIT Kharagpur)    Language Modelling: Advanced Smoothing Model    Week 3: Lecture 1    5/18

So, this is the probability for the words things let us in c times c star n, yes, c star is my nothing, but my effective n-gram count. Now, what about the words that will not seen at all that are the words that that or whenever I am saying words here it means the n terms in general. So, what about the n-grams that did not occur at all matter in data? So, from my definition, I will use the probability mass of the words that was seen once. So, what will be the probability mass? So, there are by definition there are N 1 words that was seen once. So,

each one of them occurs once. So, what is the probability mass? It is N 1 by N, yes, and this probability mass I will give to the words that was seen 0 times yes.

Now, you might have a question how much of this each of them individual, we will get. So, this is the probability mass that are the word the n-grams are did not occurs in the corpus to get that we will get, but how much of this individually each of them will get, so if you can just divided by the number of n-grams that did not occur in the corpus. So, we will see some by some example how we can be estimate this value. So, here if my count is 0, so Good-Turing estimate for the things that for the frequency c when c 0 is N 1 by n that we just saw, so that is a effective probability mass that I will give to the all the words or n-grams they did not occur in the training data. So, this gives me nice methods of estimating the probability for the words that did not occur in my training data.

(Refer Slide Time: 10:10)

*Complications*

*What about words with high frequency?*

- For small  $c$ ,  $N_c > N_{c+1}$
- For large  $c$ , too jumpy

*Simple Good-Turing*

Replace empirical  $N_k$  with a best-fit power law once counts get unreliable

$N_c \quad N_{c+1} = \dots$

Pinan Goyal (IIT Kharagpur) Language Modelling: Advanced Smoothing Model Week 3: Lecture 1 6/18

Now, there are certain complications that you might have when you applying this Good-Turing estimate. So, for a small c, generally you will see that N c is greater than N c plus 1 that is a number of words having a frequency c is greater than number of words having a frequency c plus 1. So, remember what would be empirical law that states that, so that is a (Refer Time: 10:34) law that is states that there is a universal relationship. So, this is generally good, but it might happen when we go to very, very high frequencies certain frequency are not observing that, so this is possible.

So, what do we do in general? So, when we go to higher and higher frequencies for larger k's or larger c's here we can replace it with some sort of best fit power law and we will do some normalizations. So, instead of doing it for each individual k, I might actually just fit, so I have N 1, N 2 and so on. And when I go to the higher values, I simply fit some sort of power law and that is so that it is normalized and I will obtain the same probability mass that is 1. So, this is power law can be fit it. In general, so in various toolkits that you might very you might use this method, they do it for only few values of k summation few values of k, maybe k is equal to 5 or k is equal to 10. Then the formula that we were using might have to be readjusted, but the basic idea was is same what we have seen here.

(Refer Slide Time: 12:02)

Count c	Good Turing c*
0	.0000270
1	0.446
2	1.26
3	2.24
4	3.24
5	4.22
6	5.19
7	6.21
8	7.24
9	8.25

So, now suppose you apply this Good-Turing smoothing methods for a corpus like AP Neswire academic press news, there are 22 million words and so that is why you are seeing for different counts what is the Good-Turing estimate that was seen, this is for bigrams. So, what yours observing in the table? So, for the bigrams that occurred 0 number of times the Good-Turing estimate was 0.0000270; for ones, 0.446; for twice 0.126. Now, so how do I actually get this number 0.0000270? So, we can just do that quickly.

So, remember for the words that occur zero times what is the total probability that we are giving them getting the probability of N 1 divided by N. Now, what is the probability given to the individual word? So, you multiplied by N 0; N 0 is the number of bigrams that occurring zero times. What would be the effective count? Effective count we will remove this

N, yes. So, it will be N 1 divided by N 0. Now, what is N 1, so you need to find out how many bigrams occurred once yes that you can find from training data how do you find N 0 that is how many bigrams did not occurred in my data.

Now, if you remember from Shakespeare corpus, so how do we find out how many bigrams did not occur at all. So, we need to see what is my vocabulary size, what are different unigrams. Suppose in my vocabulary size of V that is my number of unigrams. So, how many bigrams are possible? V square bigrams are possible. So, how many actually occurred? Suppose I know that out of this some N b bigrams occurred in my corpus. So, V square minus N b gives me the bigrams that did not occur at all in my corpus. So, this will become my N 0 number of bigrams that did not occurred in my corpus. So, this value I obtain by finding out N 1 number of bigrams that occur once divided by V square minus N b, N b is number of bigrams in total. So, that is the unique bigrams that occur in my corpus.

Now, what is the other observation that you might have from this table? While looking at this table, so for 0 and 1, this is different, but once you start seeing from 2 onwards, what is the pattern that you seen. You see that most of the effective counts are the original count minus some value like 0.75. So, you see all of these a roughly original value minus 0.2 to 75. So, now, one might give why do not I just use c minus 0.75 in maybe give a different value to the initial two components that is also done in practice. So, this is called the absolute discounting that is what we will see next.

(Refer Slide Time: 15:40)

Absolute Discounting Interpolation

Why don't we just subtract 0.75 (or some  $d$ )?

$$P_{\text{AbsoluteDiscounting}}(w_i | w_{i-1}) = \frac{c(w_{i-1}, w_i) - d}{c(w_{i-1})} + \lambda(w_{i-1})P(w_i)$$

We may keep some more values of  $d$  for counts 1 and 2  
But can we do better than using the regular unigram correct?

Pawan Goyal (IIT Kharagpur) Language Modelling: Advanced Smoothing Model Week 3 / Lecture 1 9 / 18

So, the idea is that why do not I subtract something like 0.75 or some d from each bigram that occur say more than once or whatever. And I can have one or two different values of d for some initial bigrams, and this is called the absolute discounting methods . And you

$$P_{\text{AbsoluteDiscounting}}(w_i | w_{i-1}) = ((c(w_{i-1}, w_i) - d) / c(w_{i-1})) + \lambda(w_{i-1})P(w_i)$$

can probably interpolated with my unigram probability that we saw in the last lecture to give a better estimate, yes, this is called linear interpolation with the unigram probability of the absolute discounting method and that is the formula that we have written here. So, this is I am doing some sort of discounting from each bigram count, yes, plus I am interpolating it with my unigram probability with weight some lambda w i minus 1. Why do I need a weight here, so that when I saw this probability absolute discounting w i given w i minus 1 for each individual w i this will add up to 1 that is why I need some sort of normalization constant that will make sure that this will add up to 1.

So, we can keep probably some values of d for count 1 and 2, and otherwise I can use these particular interpolation methods. Now, that is why the idea of our other advanced smoothing model comes that was our Kneser-Ney smoothing methods. So, what is that? So that is can we do better than using the regular unigram correct. So, when before we first started with a simple advanced smoothing, we were giving uniform weight to each particular n-gram uniform addition. Then from they moved on to unigram prior, we said why do we give uniformly we give a higher weight to the words that occur more often in my corpus.

(Refer Slide Time: 17:53)

Now, we have seen can be do better than even this and that is why we will see the Kneser-Ney smoothing idea. So, what is the idea? So, know again let us go to the Shannon game. Suppose, at this sentence, I cannot see without my reading and have to filling this blank. And suppose possibilities are glasses and Francisco and all of us will say I should fill in glasses here. Now, take a scenario where reading glasses does not occur in my training data and reading Francisco also does not occur in my training data.

Now, if I use the previous model what will I do? The probability of filling in glasses versus Francisco will depend on their unigram probabilities. And suppose this is my data from some yours corpus, so I will find probability of Francisco is higher than probability of glasses that can very well happened and that means, I will add up filling it Francisco here there is not the correct word correct choice here. So, can we do something better than simply using their unigram probabilities that is the idea of Kneser-Ney smoothing. So, Francisco is more common than glasses in my data, but can I use this idea that in my data whenever see Francisco it occurs only after ‘San’, in this occurrence San Francisco.

On the other hand, whenever I see glasses, it occurs some after multiple different words. So, this is the intuition that is I will see given a word what are the other words it comes with. So, here if I take word as Francisco, it occurs maybe with only one word, yes, only one word San, but glasses occurs with many other words right, it can occur with different, different words and this. So, both glasses are occurs with ten different words and Francisco occurs with only one different word. So, what is the intuition? If a word occurs with many other words that means, it is more likely to complete this particular bigram that we are seeing right now.

So, reading, after reading, a word is more likely to occur if it has already completed many other bigram. So, if it is more commonly used bigrams with many, many other words, this will not happen with Francisco because Francisco is used only after San. So, given a new context, probably San Francisco is not good choice, but glasses are occur many, many other different words. So, after read completely new context glasses is a better choice than Francisco.

Formally, how this method used this intuition? So, in instead of using simple probability of this word unigram probability, each probability continuation word, so that is how likely is this word to appear as a novel continuation. What do I mean by novel continuation how likely it is to complete a bigram or in n-gram? So, how do I find out the probability? So, for each

word to find this probability as we said we have to count the number of bigram types it completes, how many bigrams where it occurs as the last one - the final word. Now, for each bigram, whenever it occurs the first time, it was like a novel continuation, so that is what we mean by novel continuation probability here that how many bigrams it is complete. So, this will be simply, it will be proportional to the number of bigrams it is completing. So, here how many  $w_{i-1}$  are there such that  $w$  occurs after that in my corpus. So, that is the probability of continuation.

$$P_{continuation}(w) \propto |\{w_{i-1} : c(w_{i-1}, w) > 0\}|$$

(Refer Slide Time: 22:01)

It is a same thing, what is the probability of continuation that should depend on the number of bigram types it completes. Now, how should I normalize it? I want to find this probability for each word in the corpus, yes. So, this is proportional to the number of bigram types it completes. So, I should find out in total how many bigram types are there that each word is completing, so that is effective their number of bigram types in total not totals the number of bigram types, how many different bigram types are there. So, here also we are only taking the types, how many different bigram types we are completing.

So, I will normalize it by using all possible bigrams. So, here there says all possible bigrams because I am saying the count is greater than 0. In my training data, how many bigram types I

have seen. So, this will give me the probability continuation of the word the w. So, once I have found this probability, I can even go back to my unigram prior based absolute discounting method and replace p w i by p continuation w i and that is my Kneser-Ney smoothing model. So, instead of using the unigram priors for smoothing, I used this Kneser-Ney smoothing by using the continuation probability. So, if I do that a frequent word like Francisco occurring only in one context of San will have a low continuation probability.

]/[

$$P_{continuation}(w) = [|\{w_{i-1} : c(w_{i-1}, w) > 0\}|]$$

$$|\{w_{j-1} : c(w_{j-1}, w) > 0\}|$$

(Refer Slide Time: 23:42)

$P_{KN}(w_i | w_{i-1}) = \frac{\max(c(w_{i-1}, w_i) - d, 0)}{c(w_{i-1})} + \lambda(w_{i-1}) P_{continuation}(w_i)$

$\lambda$  is a normalizing constant

$$\lambda(w_{i-1}) = \frac{d}{c(w_{i-1})} |\{w : c(w_{i-1}, w) > 0\}|$$

So, this will give an model with Kneser-Ney smoothing. You see so all these are there is all same this is same now instead of p w i we are put p continuation w i that we get from the previous slide whatever formula we wrote in the previous slide that will give me this continuation p w i. So, now this one question here that we discuss even in the earlier example how do we find lambda w i minus 1. So, what was the intuition I gave? The hint was that lambda w i minus 1 should be such that summation over k n smoothing of w i given w i

minus 1 for all w i in my vocabulary should be 1.  
 -1)

$$P_{KN}(w_i | w_{i-1}) = \max(c(w_{i-1}, w_i) - d, 0) / c(w_i + \lambda(w_{i-1}) P_{continuation}(w_i))$$

So, now you take it as a simple exercise that suppose I want to find out what will be the value of lambda w i minus 1 for this particular formula. And you can assume that d lies between 0 to 1.

For this particular

$$\lambda(w_{i-1}) = (d/c(w_{i-1})) |\{W : c(w_{i-1}, w) > 0\}|$$

case, what is the value of lambda? Lambda w i minus 1 that I will get, so answer is also on this slide. So, you will get lambda w i minus 1 of this. So, try to find out if I have to satisfy the constant of the summation of the probabilities for all the words should add up to 1, what is the value of lambda that I end up with that will give you this value - particular value.

(Refer Slide Time: 25:21)

*As N increases*

- The power (expressiveness) of an N-gram model increases
- But the ability to estimate accurate parameters from sparse data decreases (i.e. the smoothing problem gets worse).

A general approach is to combine the results of multiple N-gram models.

Pawan Goyal (IIT Kharagpur)    Language Modelling: Advanced Smoothing Model    Week 3: Lecture 1    13 / 18

So, other than these two models of Good-Turing and Kneser-Ney smoothing, there are other smoothing models that you might be able to use. So, one simple idea instead of using a new smoothing model why do not we combine various models; so remember we always doing that somewhere. So, you computing bigram, I was trying to use the probability of the unigram, for unigram priors smoothing, now can I generalized this idea. Suppose, I am finding out the trigram probability, trigram language model; now from my data I can compute the trigram

language model probabilities bigram and unigram using the MLE. Can I try to effectively combine these different probability models to find my trigrams language model?

So, there are actually two very, very popular ways in which this can be done one is called interpretation based methods, another is called back off of language models. So, we will just see some simple intuition how this can be combined together. So, what we have seen as I increased my N power of my N-gram model increases we also see saw that in my when I saw the perplexity values. When I take a larger N-gram, I get lower perplexity I get better perplexity that means, the model becomes much better when I take a larger N. But what is the problem that happens if I take a larger N, remember why we did all this is smoothing because when I am taking larger N, the data becomes more and more sparse. So, many of the actual N-grams become 0, the probability become 0.

So, at one side higher N-gram is better, a trigram model is better other side is very, very sparse. So, why cannot I take advantage of the both things that is I take a higher order N-gram model whenever I have the data, but whenever I do not have the data I try to use the low N-gram model. So, I do that togetherness single model, so that is the basic idea of these model that we will see. So, now general approach is can be combined multiple N-gram models to get a single model that can also take the advantage of larger N-gram model, but avoid this sparseness problem.

(Refer Slide Time: 27:44)

*It might help to use less context*  
when you haven't learned much about larger contexts

**Backoff**

- use trigram if you have good evidence
- otherwise bigram, otherwise unigram

**Interpolation**

mix unigram, bigram, trigram

Pawan Goyal (IIT Kharagpur)   Language Modelling: Advanced Smoothing Model   Week 3: Lecture 1   14 / 18

So, that is some cases might help to use less context whenever you do not have information about larger data, larger context. So, in back off what you will do suppose I am computing a trigram language model, so  $w_i$  given  $w_{i-1}$ ,  $w_{i-2}$  whenever this trigram occurs in my training data I will use the probability as for my MLE. But if it does not occur in my training data I will back off to my bigram language model so that means, whenever the count of  $w_{i-1}$ ,  $w_i$  is greater than 0, I will go to bi bigram; suppose that is also 0 then I will go to unigram. This is idea of back off language model. So, if you have good evidence you go to trigram, otherwise you go to bigram, otherwise you go to unigram. On the other hand, interpolation you compute all three and just interpolate them together you try to mix these together.

(Refer Slide Time: 28:46)

*Backoff*

*Estimating  $P(w_i|w_{i-2}w_{i-1})$*

- If we do not have counts to compute  $P(w_i|w_{i-2}w_{i-1})$  estimate this using the bigram probability  $P(w_i|w_{i-1})$
- If we do not have counts to compute  $P(w_i|w_{i-1})$ , estimate this using the unigram probability  $P(w_i)$

$P_{bo}(w_i|w_{i-2}w_{i-1}) =$

- $\hat{P}(w_i|w_{i-2}w_{i-1})$ , if  $c(w_{i-2}w_{i-1}w_i) > 0$
- $\lambda(w_{i-1}w_{i-2})P_{bo}(w_i|w_{i-1})$ , otherwise

where  $P_{bo}(w_i|w_{i-1}) =$

- $\hat{P}(w_i|w_{i-1})$  if  $c(w_{i-1}w_i) > 0$
- $\lambda(w_{i-1})\hat{P}(w_i)$ , otherwise

Pawan Goyal (IIT Kharagpur) Language Modelling: Advanced Smoothing Model Week 3: Lecture 1 15/18

So, let us see what do we will do in back off language models. So, idea is that I am computing a trigram model  $w_i$  given  $w_{i-2}$ ,  $w_{i-1}$ . So, here this is the previous word and this is previous to previous word. So, if I do not have counts for  $w_{i-2}$ ,  $w_{i-1}$ ,  $w_i$ , suppose this is 0 then I use probability  $w_i$  given  $w_{i-1}$ . Now, suppose count of  $w_{i-1}$ ,  $w_i$  is also 0, then you go to probability  $w_i$ . Now, so you might add up having some problems with how do I normalize the final probability distribution that I will get. So, this is the particular formula, recursive formula of back off that you can use. So, back off for this trigram model  $w_i$  given  $w_{i-2}$ ,  $w_{i-1}$  is  $\hat{P}(w_i|w_{i-2}w_{i-1})$  if  $c(w_{i-2}w_{i-1}w_i) > 0$  if the count is greater than 0.

$$P_{bo}(w_i|w_{i-2}w_{i-1}) = \hat{P}(w_i|w_{i-2}w_{i-1}), \text{ if } c(w_{i-2}w_{i-1}w_i) > 0$$

Now, what is  $\hat{P}$ ? This is actually similar to my maximum likelihood estimate that you will get, but it has been something has been subtracted from there. So, some probability mass has been shown from there, so there it when it can be given to the other counts. Why we need that suppose you are computing this trigram back off probability model. So, in general you are doing it after  $w_i - 2$ ,  $w_i - 1$ , you are finding for various words  $w_1, w_2, w_3$  suppose this occurs three times, this occurs two times, this occurs 0 times.

$$P_{bo}(w_i | w_{i-1}) =$$

$$\hat{P}(w_i | w_{i-1}), \text{ if } c(w_{i-1}w_i) > 0$$

So, now, once use the MLE method estimate immediately this will have a probability of 3 by 5, this I have probability of 2 by 5, suppose there is no other bigram and this will have a probability of 0. And now you want to use back off to use probability  $w_3$  given  $w_i - 1$ , but here itself probability mass is adding up to 1. So, how can you use the bigram probability? So, for that you might have to reduce some mass from these two values, so that is why we are writing  $\hat{P}$  and there are different ways you can reduce. So, we will take someone simple example we are doing it by some simple constant, you are reducing some simple constants, but this can be proportional also you can multiply some (Refer Time: 31:35). So, you will take that is my  $\hat{P}$ , it is a reduced probability value from my MLE estimate.

So, now, if the count is greater than 0, I take  $\hat{P}$ ; if the count is equal to 0, then I have to use the previous I have to back off to the bigram model. So, I have to use probability  $w_i$  given  $w_{i-1}$  and that is where I back off to this model, but what is interesting that you have seen here. So, this is a recursive definition I am using the back off probability of  $w_i$  given  $w_{i-1}$ . And I multiplying with some constant again to ensure that the probability mass adds up to 1. So, the recursive definition, so how do we define probability  $w_i$  given  $w_{i-1}$ , again it is  $\hat{P}$  if the count is greater than 0; and  $P_{bo}(w_n)$  if the count is equal to 0. So, I think it will help, if you take a simple example and try to see how do we actually use this definition to compute the back off probabilities.

(Refer Slide Time: 32:55)

The slide has a blue header bar with the title "Example Problem". The main content area contains the following text:

In a corpus, suppose there are 4 words,  $a$ ,  $b$ ,  $c$ , and  $d$ . You are provided with the following counts.

n-gram	count	n-gram	count	n-gram	count
<u>aba</u>	4	<u>ba</u>	5	a	8
<u>abb</u>	0	<u>bb</u>	3	b	9
<u>abc</u>	0	<u>bc</u>	0	c	8
<u>abd</u>	0	<u>bd</u>	0	d	7

Use the recursive definition of backoff smoothing to obtain the probability distribution,  $P_{\text{backoff}}(w_n | w_{n-2} w_{n-1})$ , where  $w_{n-1} = b$  and  $w_{n-2} = a$ .  
Also assume that  $\hat{P}(x) = P(x) - 1/8$ .  $P_{\text{bo}}(w | ab)$

At the bottom of the slide, there is a footer bar with the following information:

Pawan Goyal (IIT Kharagpur) Language Modelling: Advanced Smoothing Model Week 3: Lecture 1 / 16 / 18  
7:40 PM 10/4/2016

So, let us take an example. So, here we are taking a corpus, where we are having only four words  $a$ ,  $b$ ,  $c$  and  $d$  and this is one data that is provided. So, what is this data. So, it is telling how many times  $a$  occurs after  $ab$ , it occurs four times; on the other hand,  $b$  occurs after  $ab$  - zero times,  $c$  occurs after  $ab$  - zero times. Similarly,  $a$  occurs after  $b$  - 5 times,  $b$  occurs after  $b$  - 3 times,  $c$  occurs after  $b$  - 0 times, and also the individual word  $a$  occurs 8 times,  $b$  occurs 9 times,  $c$  occurs 8 times,  $d$  occurs 7 times.

Now, given this data, you are asked to compute the back off probability model of  $w_n$  given  $w_{n-1}$  where the previous word is  $b$ , and previous to previous word is  $a$ . And you have also told that you can use this simple definition for  $\hat{P}$  that is  $P(x) - 1/8$ . Suppose, I want to use that to find out the back off probability distribution, so that is nothing but probability back off of word given  $a$  and  $b$ ;  $a$  is previous to previous word, and  $b$  is the previous word.

(Refer Slide Time: 34:31)



$$P_{bo}(w|ab) = \begin{cases} a \rightarrow 4 \rightarrow \hat{P}(a|ab) \\ b \rightarrow 0 \rightarrow \lambda_{ab} \cdot P_{bo}(b|b) \\ c \rightarrow 0 \rightarrow \lambda_{ab} \cdot P_{bo}(c|b) \\ d \rightarrow 0 \rightarrow \lambda_{ab} \cdot P_{bo}(d|b) \end{cases}$$

$$\hat{P}(a|b) = \frac{5}{8} - \frac{1}{8} = \frac{4}{8}$$

$$P_{bo}(w|b) = \begin{cases} a \rightarrow 5 \rightarrow \hat{P}(a|b) = \frac{5}{8} \\ b \rightarrow 3 \rightarrow \hat{P}(b|b) = \frac{3}{8} - \frac{1}{8} = \frac{2}{8} \\ c \rightarrow 0 \rightarrow \lambda_b \cdot \hat{P}(c) = \\ d \rightarrow 0 \end{cases}$$

So, let us try to do that. So, I want to find out the back off probability  $w$  given  $a, b$ . So, this would be  $w$  can take  $a, b, c$  and  $d$ . So, right now from my table, what are the counts, I am seeing  $a$  occurs, so this occurs four times, this occurs zero times, this occurs zero times, this occurs zero times. So, assumed by definition of back off probability, I can say that this probability will be  $\hat{P}(a|ab)$ , because the count is greater than the 0, but this will be  $\lambda_{ab}$  times previous two words  $ab$ . So, this  $\lambda_{ab}$  a constant times back off probability of so I will go to the previous word back off probability of  $b$  given  $b$ , this will be  $\lambda_b$  times back off back off probability of  $c$  given the previous word  $b$ ; and this will be  $\lambda_b$  a  $b$  back off probability of  $d$  given  $b$ .

So, what it say is that now you have to compute the back off probability of  $b$  given  $b, c$  given  $b$ , and  $d$  given  $b$ . So, again we use the definition. So, now, I am computing back off probability of  $w$  given  $b$ , and  $w$  is equal to  $a, b, c$  and  $d$ . Now, I see  $a$  occurs 5 times,  $b$  occurs 3 times,  $c$  occurs 0 times,  $d$  occurs 0 times. So, as per my definition this will be  $\hat{P}(a|b)$  what will be that that will be  $5$  by  $8$ , yes minus  $1$  by  $8$  that is  $4$  by  $8$ . What will be this  $\hat{P}(b|b)$  given  $b$   $3$  by  $8$  minus  $1$  by  $8$  that is  $2$  by  $8$ ? Now, what will be this? This occurs zero number of times. So, this will be  $\lambda_b$  of  $b$  times unigram probability of  $c$  maybe  $\hat{P}(c|b)$ . Now, what is this? We can see from my data what is  $\hat{P}(c|b)$ . So,  $c$  occurs a times, yes and total number of, so what is the total of my unigram count, if you will add up to 32.

(Refer Slide Time: 37:19)

$$\begin{aligned}
 P_{bo}(w|ab) &= \left[ \begin{array}{l} b \rightarrow 0 \rightarrow \lambda_{ab} \cdot P_{bo}(b|b) \\ c \rightarrow 0 \rightarrow \lambda_{ab} \cdot P_{bo}(c|b) \\ d \rightarrow 0 \rightarrow \lambda_{ab} \cdot P_{bo}(d|b) \end{array} \right] \\
 P_{bo}(w|b) &= \left[ \begin{array}{l} a \rightarrow 5 \rightarrow \hat{P}(a|b) = \frac{5}{8} - \frac{1}{8} = \frac{4}{8} \\ b \rightarrow 3 \rightarrow \hat{P}(b|b) = \frac{3}{8} - \frac{1}{8} = \frac{2}{8} \\ c \rightarrow 0 \rightarrow \lambda_b \cdot \hat{P}(c) = \lambda_b \cdot \left[ P(c) - \frac{1}{8} \right] \\ d \rightarrow 0 \rightarrow \lambda_b \cdot \hat{P}(d) = \lambda_b \cdot \frac{\frac{1}{8}}{32} \end{array} \right] \\
 \lambda_b \left( \frac{1}{8} + \frac{3}{8} \right) &= \frac{4}{8} \\
 \lambda_b \cdot \left[ \frac{3}{8} \right] &= \frac{4}{8} / N \\
 \lambda_b &= \frac{4}{8} / N
 \end{aligned}$$

So, this p c the MLE varies, so it is lambda b times p c minus 1 by 8, p c is 8 by 32 yes. So, this will become 8 by 32 - 1 by 4 minus 1 by 8 that will be lambda b times 1 by 8. Similarly, for d is equal to 0, I can write lambda b times P hat d that is that is lambda b times p d minus 1 by 8 that is lambda b times p d here was 7 by 32 minus 1 by 8. So, that would be lambda b times this will be 7 minus 4 3 by 32.

So, suppose you get these values, now how will you go to the probability P w given a b you have to first find what is the back off probabilities of for these two cases c and d. So, for that you define the what is the value of lambda b, now how do you find lambda b if you normalized this right. So, you will say lambda b times 1 by 8 plus 3 by 32 plus 4 by 8 plus 2 by 8 should give me 1, right this should be 1 if you sum over all the words. So, what this is? This say lambda b times 7 by 32 is equal to 1 minus 6 by 8 that is 2 by 8. So, what is lambda b? So, we will get, you will get this 8 by 7.

So, once we substitute add up this 8 by 7 here that will give you this probability 8. So, you will find this probability and this probability also. So, you have the back off probability for this. Now, put this back off probability here, and we have a constant how do you find this constant, again you will have to normalized it lambda a b times summation of this plus this probability should add up to one that will give you my lambda a b and also give you this probability distribution. So, that is why you find the back off probability distribution using this recursive definition.

(Refer Slide Time: 40:18)

*Linear Interpolation*

*Simple Interpolation*

$$\tilde{P}(w_n | w_{n-1} w_{n-2}) = \lambda_1 P(w_n | w_{n-1} w_{n-2}) + \lambda_2 P(w_n | w_{n-1}) + \lambda_3 P(w_n)$$
$$\sum_i \lambda_i = 1$$

*Lambdas conditional on context*

$$\tilde{P}(w_n | w_{n-1} w_{n-2}) = \lambda_1(w_{n-2}, w_{n-1}) P(w_n | w_{n-1} w_{n-2}) + \lambda_2(w_{n-2}, w_{n-1}) P(w_n | w_{n-1}) + \lambda_3(w_{n-2}, w_{n-1}) P(w_n)$$

Pawan Goyal (IIT Kharagpur)    Language Modelling: Advanced Smoothing Model    Week 3: Lecture 1    17 / 18

And what do you do linear  
interpolation, we

$$P(w_n | w_{n-1} w_{n-2}) = \lambda_1 P(w_n | w_{n-1} w_{n-2}) + \lambda_2 P(w_n | w_{n-1}) + \lambda_3 P(w_n)$$

have different unigram, bigram, trigram models, I try to interpret them in different proportions. So, what we are doing here? We have computed a trigram model, bigram model, unigram model and different proportions lambda 1, lambda 2, lambda 3 are given to this, so such that they will add up to 1. In general you can also conditional lambdas on the context, what is your previous and previous-to-previous word the lambda might depend on that. So, this is the same equation except that now lambda is a depending on the context.

(Refer Slide Time: 41:02)

*Setting the lambda values*

*Use a held-out corpus*

Choose  $\lambda$ s to maximize the probability of held-out data:

- Find the N-gram probabilities on the training data
- Search for  $\lambda$ s that give the largest probability to held-out data

Pawan Goyal (IIT Kharagpur)    Language Modelling: Advanced Smoothing Model    Week 3: Lecture 1    18 / 18

And now the question might be how do I choose a held-out corpus, how do I choose my lambdas, so that you will do as we say earlier you will have an held-out corpus, you will find out which values are lambda gives you the highest probability and that that lambdas you can has for your smoothing. So, that were we finish our topic of language model. So, we covered lot of simple smoothing methods, and some advanced smoothing methods and then we will now go to the next topic of so we will start with the topic of morphology, so that will be the topic of the next lecture.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology Karagpur**

**Lecture – 13**  
**Computational Morphology**

Hello everyone, so welcome back for the second module of third week. So, in the last week in the last module we were discussing about language modeling that is how we can effectively use the word ordering information for where is applications. We will go to the next topic, we will talk about the morphology information that what are the different modules that help us captures information and what are different linguistic terms that are involved. So, today we will start with the linguistic terms and then we will end with what are possible modules that can help us capture this in an automatic manner.

(Refer Slide Time: 01:00)

**Morphology**

Morphology studies the internal structure of words, how words are built up from smaller meaningful units called **morphemes**

**dogs**

- 2 morphemes, 'dog' and 's'
- 's' is a plural marker on nouns

**unladylike**

3 morphemes

- un- 'not'
- lady 'well-behaved woman'
- -like 'having the characteristic of'

Pawan Goyal (IIT Kharagpur)      Computational Morphology      Week 3: Lecture

So, we are talking about computation morphology today, so what is morphology? In morphology we study what is internal structure of words. So, that is given a particular word in a language, what are the different meaningful units it is made up of and each small unit is called a morphemes. So, let us take some examples, so if I take the word dogs, so this is made up of two different units; one is dog the actual root word and another is affix that is applied to the word to make it a plural that is s. So, there are two morphemes in this single word dogs.

Similarly if I take a word like unladylike. So, we are having three different units here, so these are three different morphemes. So un; that corresponds to not, a negation or opposition, then lady well behaved women like that is having the characteristic of. So, there are three different morphemes that together constitute this single word.

(Refer Slide Time: 02:00)

Allomorphs

Variants of the same morpheme, but cannot be replaced by one another

*Example*

- opposite: un-happy, in-comprehensible, im-possible, ir-rational

Pawan Goyal (IIT Kharagpur) Computational Morphology Week 3: Lecture 2 3 / 19

So, now we will also see what are the various linguistic notions that you might want to be aware of when we are talking about morphology, so for example, what are allomorphs? So given a word like happy, if you have to convert it into its opposites; so we are making a word like unhappy, but if I take another word like rational, I will use i r; so I will make irrational. So, if they are different morphemes that can be used for the same purpose they are called allomorphs, so here is an example.

So allomorphs; we are saying variants of the same morphemes, but in general we cannot replace by one another. So, let us see here, so if I have the word happy; I can use u n, to make it unhappy, comprehensible; I can use i n, incomprehensible, if possible I can use i m; impossible and rational I will say irrational, but you can see that we cannot replace I r by u n here. So, I cannot say un rational; that is not a valid English word, so there are various reasoning for why a particular morpheme is used one of these is could be because of the phonemes that are there in the particular word. So, irrational you have the irrational with happy, we have unhappy, so for different words you have different sort of morphemes that you apply to make opposites. So, that is for one particular function, but

for different functions like tenses and person and all that, you again have various categories of morphemes that are applied.

(Refer Slide Time: 03:36)

**Bound**  
Cannot appear as a word by itself.  
-s (dog-s), -ly (quick-ly), -ed (walk-ed)

**Free**  
Can appear as a word by itself; often can combine with other morphemes too.  
house (house-s), walk (walk-ed), of, the, or

Pawan Goyal (IIT Kharagpur)      Computational Morphology      Week 3: Lecture 1

So, again when we talk about morphemes there is a distinction called bound morphemes versus free morphemes. So what are those? So bound morphemes are those morphemes that cannot appear as a word by itself. So, for example take the previous example unhappy; so I cannot take the morpheme un here and use it as a unique single word itself. So, this is bound by another word that is applied with this say unhappy, so it is bound with happy or some other words with this you can be use it.

On the other hand, if I take the word happy it is free morphemes, it can use on it is soon that is how we divide the morphemes into these two categories bound verses free. So, here are some examples, so bound morphemes are like s; so it can be used for making plural. So, with dog we have dogs; l y, so quick to quickly and e d; walk to walked. So, these are all bound morphemes and then there are free morphemes when they can appear as a word by themselves and they can also combine with other morphemes if needed. So, for example, if I take the morphemes like house, it can appear as a word by itself, but can also be combine with other morphemes like s to make plural houses. Similarly walk; it can combined with e d to make a past tense of walked and so on. Some words like of, in etcetera they are also free morphemes, they can appear on their own.

(Refer Slide Time: 05:05)

**Stems and Affixes**

- Stems (roots): The core meaning bearing units
- Affixes: Bits and pieces adhering to stems to change their meanings and grammatical functions

Mostly, stems are free morphemes and affixes are bound morphemes

Pawan Goyal (IIT Kharagpur) Computational Morphology Week 3: Lecture 2 5 / 19

So, then there is another distinction and this is very very important when we talk about morphology. So in general when I talk about a word, it can have two main parts; one is the stem the main the root word and another is the affix and it can have more than one of affixes as well, so what are stem and affixes. So, when we talk about a word, so a stem is the core meaning bearing part. So, I take a word like boys, so the stem here will be boy. So, that is the meaning can bearing part and then there are various bits and pieces that can be applied to it that can ultra certain grammatical functions.

So, I can apply and as at the end of to make a plural of this, so these are called affixes. So, in general you can make a correspondence between stems and affixes and bound and free morphemes. So, you can say that the stems are kind of free morphemes and the affixes are kind of bound morphemes, so this kind of correspondence you can make.

So, now what are the different types of affixes that you can apply with the given stem? So we just discussed one simple example of applying s. So, this s apply after the word boy are make a plural with boys; so this is called suffix that is applied at the end of the word, but in general what are the different types of affixes that can be applied.

(Refer Slide Time: 06:40)

*Types of affixes*

- Prefix: un-, anti-, etc (a-, ati-, pra- etc.)  
*un-happy, pre-existing*
- Suffix: -ity, -ation, etc (-taa, -ke, -ka etc.)  
*talk-ing, quick-ly*
- Infix: 'n' in '*vindati*' (he knows), as contrasted with *vid* (to know).  
*Philippines: basa 'read' → b-um-asa 'read'*  
*English: abso-bloody-lutely (emphasis)*
- Circumfixes - precedes and follows the stem  
*Dutch: berg 'mountain', ge-berg-te 'mountains'*

Pawan Goyal (IIT Kharagpur) Computational Morphology Week 3: Lecture 1

So, we start with prefixes, so prefixes are applied before the word. So, some example in English are un, so we make unhappy, anti; antinational, pre; preexisting and so on. We have these prefixes in other languages also for example, in Hindi and Sanskrit; we have the prefixes like [FL] that we can apply with various words or verbs, so these are of prefixes.

Then we have suffixes that are applied after the word, so like with the word talk; I can make a new word talking by applying the suffix by i n g. Similarly the word quick; I can make a new word quickly by applying the suffix l y; at the end and in the case of Hindi for example, you can have this exercise like [FL] and [FL] etcetera that can apply after the words.

Then, so the prefix and suffix are the major suffixes, affixes that we have in language, but certain language have some other kind of exercise also. So, one is called infix that is it is applied in between that stem not before or after that, so example is like in Sanskrit we have the word with to know and we make the present tense by putting vindati. So, the character n; the phon in n is applied in between; between v and the, so it becomes vindati. So, there are examples in other languages also like in the case of philippines here basa ; b a s a is read and to make it; converted to the past tense read, the infix u m is applied in between, so we have b u m a s a , so u m is applied in between is called infix.

Now, so when you think of an infix in the case of English; English it is not common; in common language it is not infix is not used, but in general can we make a word in English that that uses infix. So, here an example is if I take the word absolutely and put a word bloody inside, so you can have a word like absolutely; bloodilutely. So, this is to put some emphasis that is not used in common English, but in certain movie dialogues and all you can find such kind of words.

Finally, so what you think can be the fourth category, we have prefix, suffix, infix; what is the meaning is that a suffix, an affix comes before as well as after, so this is called circumfix. So, this proceeds as well as follows the same word, so an example in Dutch; so you have a word for mountain berg and to convert it to plural we have an affix that is applied before g e as well as after. So, the whole affix is g e t e half of that is applied before and half of that is applied after, so this is called circumfix. So among this four kind of affixes, the first two are very very common prefix and suffix and the last two are expressive to certain languages only.

(Refer Slide Time: 09:54)

*Content and functional morphemes*

*Content morphemes*  
Carry some semantic content  
*car, -able, un-*

*Functional morphemes*  
Provide grammatical information  
*-s (plural), -s (3<sup>rd</sup> singular)*

Pawan Goyal (IIT Kharagpur)      Computational Morphology      Week 3: Lecture 1

Now, morphemes can also be divided into some other categories like content verses functional morphemes. So, what are content morphemes that are those that will that will be a semantic meaning? So, for example, if even if I take any free morphemes like car a word, it is always a content morpheme; it contains some semantic meaning. Similarly I

can take a morpheme bound morpheme like able also that is a semantic meaning that given a word, it keeps a sense of being able to do something.

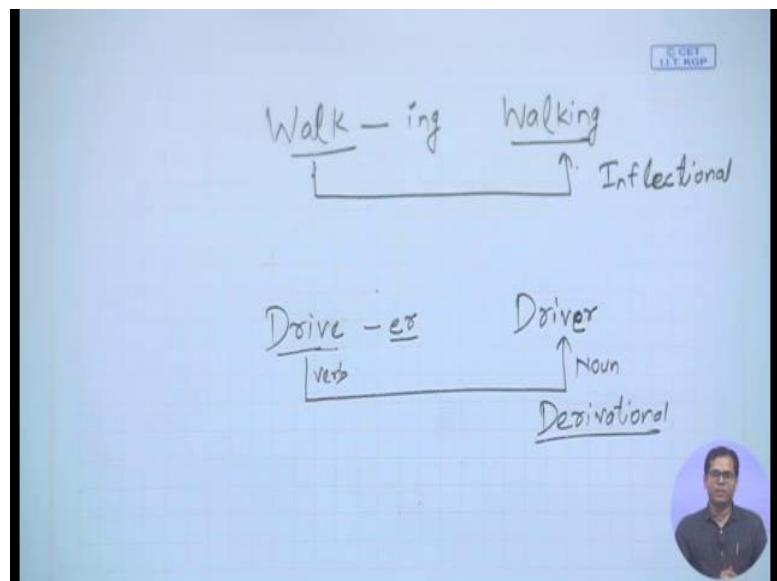
So, on the other hand there are certain morphemes that are functional that do not take the semantic content, they are only used for certain grammatical functions like as for plural, as for third singular they are both functional morphemes; you apply it after a word and it has a particular grammatical meaning, but it may not be a semantic; it may not contain a semantic meaning as such.

(Refer Slide Time: 10:52)

The slide has a blue header bar with the title 'Inflectional and Derivational Morphology'. The main content area is white with black text. It starts with a statement: 'Two different kind of relationship among words'. This is followed by two sections: 'Inflectional morphology' and 'Derivational morphology'.  
**Inflectional morphology**  
Grammatical: number, tense, case, gender  
Creates new forms of the same word: *bring, brought, brings, bringing*  
**Derivational morphology**  
Creates new words by changing part-of-speech: *logic, logical, illogical, illogicality, logician*  
Fairly systematic but some derivations missing: *sincere - sincerity, scarce - scarcity, curious - curiosity, fierce - fiercity?*

Now so based on whatever kind of affixes that you apply to a word, you will generate a new word.

(Refer Slide Time: 11:07)



So, for example, if I have a word walk and I apply a suffix i n g; I get a new word like walking. So, there is a process that converts a word walk to walking, so by putting certain morphology here; now take another example I take a verb like drive and I add some affix and make a new word like driver, so now what is the difference that you see in the two processes. So, we can call it may be e r; drive plus e r that gives you driver, so from drive we can creating driver verses from walk you are creating walking. So, what is the difference that you see in the two processes, so if you think about it; in the first case we not changing the grammatical category of the word as such.

So, this is a verb walk, it remains the verb here also walking but we add some grammatical information as a continuous text. On the other hand here, you start with the verb drive and we end up making a noun, you covert the category of the verb itself by this process of morphology. So, there are two different kind of morphological processes, so this is called inflectional morphology and this is called derivational morphology. So, so in this slide we are seeing the difference between these. So, they give you the relation between two words that you have created; first case you have created from walk to walking, so what is the relation between these two words; second case you created drive to driver; what is the relation between these two words.

So in flectional morphology it makes changes in the terms of numbers, tens, case and gender. So, for example, if you start with the word verb bring you can alter the case, you

can have brought, you can have third person singular brings and so on. So, they do not change the category of this word so, but you add some more grammatical information that is all. So, this is called inflectional morphology, you are adding certain inflectional information only in the grammatical sense that is not change in the category of the word. On the other hand, you have derivational morphology where you create some new words and you also change the category of the word. So, this is called part of speech that we will take up in the same week in the fourth module.

So, it changes the category of the word, so for example, from logic you can make logical and illogical, logicality, logician and all that and there you can see the do not have the same category all these words, they have they are different categories one is noun, adverb and so on adjective. In general derivational morphology is also fairly systematic like inflectional morphology, but sometimes certain derivations will be missing. So, for example, here is some nice examples, so if I take the word sincere I can make sincerity, scarce - scarcity, curious - curiosity, but from fierce you should not use like fiercity; although it is very very; if it looks very very regular, if you starts seen from the previous words. So, they are pretty regular, but some words do not have the corresponding derivational words.

(Refer Slide Time: 14:46)

*Morphological processes*

*Concatenation*

Adding continuous affixes - the most common process:

- hope+less, un+happy, anti+capital+ist+

Often, there are phonological/graphemic changes on morpheme boundaries:

- book + s [s], shoe + s [z]
- happy +er → happier

Now so we have talked about whatever the various kind of morphemes and what are the process, what are the different types of affixes that you can apply, but in general what are

the various morphological process that are involved to convert one word into another word, so we will talk about this. So, one simple process is called concatenation, so we will take various morphemes and concatenate together to make a new morpheme. So, for example, happy; un you make unhappy simple concatenation. Similarly here hope and less together make hopeless and un happy make unhappy and anti capital; i s t and s these four morphemes if you concatenate together and then make a single word or singular morpheme.

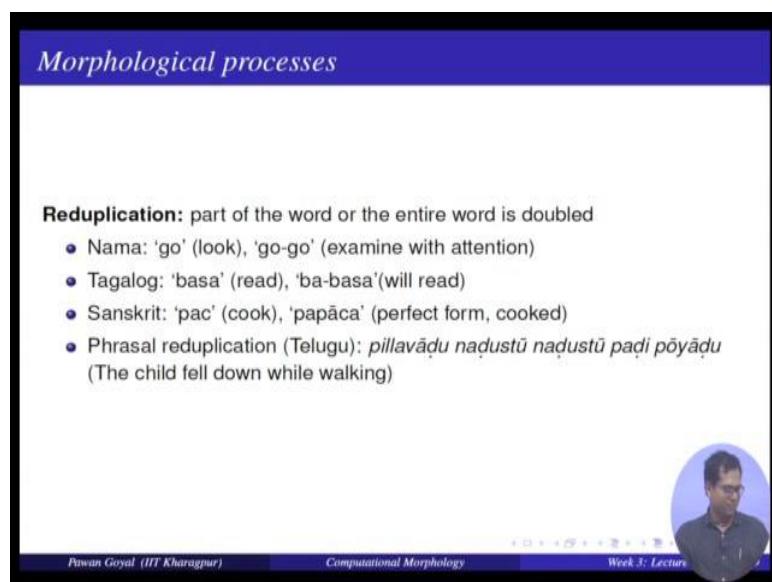
So, now then you are combining the different morphemes together; at the boundary or when they are combined there may be certain changes. The changes can be in the way the final word is pronounced or in the way the final word is written. So, there are two types called fundamic changes and graphemic changes; graphemic in the terms how the word is written and phonemic in the way word is pronounced. So, here some examples, so if I take a word book and if morpheme book and a morpheme s; if I combine together I get books. So, see the pronunciation if I take shoe and s; I get shoes, so here it is not s, it is za; it is a different phonems that terms. So, there are certain changes that happen at the boundary in terms of how the word is pronounced. Similarly if I take word like happy and er; these change at the graphemiclable. So where in the grapheme y changes to i, so this is also called simply spelling change at the morpheme boundary, so this can happen.

(Refer Slide Time: 16:47)

*Morphological processes*

**Reduplication:** part of the word or the entire word is doubled

- Nama: 'go' (look), 'go-go' (examine with attention)
- Tagalog: 'basa' (read), 'ba-basa' (will read)
- Sanskrit: 'pac' (cook), 'papāca' (perfect form, cooked)
- Phrasal reduplication (Telugu): *pillavāḍu naḍustū naḍustū paḍi pōyāḍu*  
(The child fell down while walking)



Pawan Goyal (IIT Kharagpur)      Computational Morphology      Week 3: Lecture

There are some morphological processes also like reduplication. So, then we adding in a suffix you might do reduplication somewhere in the stem. So, example is like in number language if I take a word go or look; if I want to say examine with attention, I will just repeat the go; similarly in tagalog; you have basa; for read and if I want to say will read I will read duplicate ba ba ba sa, so you reduplicating one part. In Sanskrit again this is very common phenomena or reduplication; I take the word like much that is to cook and if I want to convert it to the perfect form cooked, I will reduplicate the pa and I say papaca. So, you see the word pa is reduplicated they can be reduplication also in certain languages also example in Telugu, so certain phrase is repeated again. So, for example, I want to say the child fell down while walking, so here I say [FL]. So, here [FL] is being repeated twice for saying while walking.

(Refer Slide Time: 18:08)

*Morphological processes*

*Suppletion*  
'irregular' relation between the words  
*go - went, good - better*

*Morpheme internal changes*  
The word changes internally  
*sing - sang - sung, man - men, goose - geese*

Pawan Goyal (IIT Kharagpur)      Computational Morphology      Week 3: Lecture 2      11 / 19

There are other morphological like suppletion, where a word is completely replaced by something that has no connection at the surface label. So, I have a word like go and I am converting into the past tense and it becomes went. So, you cannot find any connection go and went and this are surface label; this is simple suppletion. Similarly with good I can make better and there are many examples in English for this, sometimes there are some internal change in the morpheme. So, from sing you want convert the plural you will; so from sing you can get sang and sung for different tenses and from men to converted to plural and you get m e n and from goose; you get geese for converting to

plural. So, you see their changes that are internally in the word, you not adding any new suffix.

(Refer Slide Time: 19:11)

*Word Formation*

*Compounding*

Words formed by combining two or more words

Example in English:

- Adj + Adj → Adj: bitter-sweet
- N + N → N: rain-bow
- V + N → V: pick-pocket
- P + V → V: over-do

*Particular to languages*

room-temperature: Hindi translation?

Pawan Goyal (IIT Kharagpur) Computational Morphology Week 3: Lecture

Then while we are talking about word formation, there is another process called compounding. So, that is I can take two different words and make a compound out of that and this compounding can be from various part of speech. For example, in English I can take two adjectives, so bitter and sweet; I can make a single compound bitter sweet, I can take two nouns rain and bow; I can make a noun rainbow, I can take a noun and verb pick and pocket and this becomes a verb pickpocket, similarly over do and what is interesting is that these compounds are very particular to languages.

So, let us take an example like room temperature that is a compound in English, but is there a corresponding compound in Hindi. So, you never say [FL] as a single compound you will say [FL] so; that means so finding this compound is an interesting problem for going to application like machine translation. So, where you cannot just directly convert the words into the equivalent target languages, so you cannot say room with combined temperature with [FL], you need to find out whether there is a compound there is relation between the two words and according to the translation.

(Refer Slide Time: 20:36)

**Word Formation**

**Acronyms**  
laser: Light Amplification by Stimulated Emission of Radiation

**Blending**  
Parts of two different words are combined

- breakfast + lunch → brunch
- smoke + fog → smog
- motor + hotel → motel

**Clipping**  
Longer words are shortened  
doctor, laboratory, advertisement, dormitory, examination, bicycle, refrigerator

Pawan Goyal (IIT Kharagpur) Computational Morphology Week 3: Lecture

Then there is another process that is called acronyms for example, the word laser is an acronym then there is blending, so your take. So, this is again a very common process linguistic, so where you take two words and then you combine together, blend them together to make a new word, but the you are taking from both the words, but you not taking the full words. So, example is breakfast and lunch you can take together and make a new word like brunch.

Similarly smoke and fog, you make a new word a smog; motor and hotel make a word like a motel and there is a process of clipping. So, that (Refer Time: 21:13) used a lot for example, I have doctor becomes doc, laboratory becomes lab and advertising becomes add, dormitory becomes dorm and examination becomes exam, bicycle becomes bike and refrigerator becomes fridge, so this is clipping; so long words are shortened

(Refer Slide Time: 21:34)

Processing morphology

- Lemmatization: word → lemma  
saw → {see, saw}
- Morphological analysis : word → setOf(lemma +tag)  
saw → { <see, verb.past>, < saw, noun.sg> }
- Tagging: word → tag, considers context  
Peter saw her → { <see, verb.past> }
- Morpheme segmentation: de-nation-al-iz-ation
- Generation: see + verb.past → saw

Pawan Goyal (IIT Kharagpur) Computational Morphology Week 3: Lecture

So, this is all the different processes that happens in happen in morphology now, so from the NLP prospective. So, we will talk about how do we process of morphology, so what do a; what does it and until what processing of morphology. So, what are different things that are done, so one simple thing is lemmatization that is given a word can identify what is the root word what is the lemma; an example is if you give me a word like saw; can I tell what is the root word, if it is a verb the root word what will be see, but if it is a noun the root word will be saw itself.

So, you can find what is the root word silly; it is lemma. There is a morphological analysis also. So, that is even a word can I find out what is the corresponding lemma along with morphological category of that word, so it is particular tag. So, example is I take a verb word like saw; can I tell the lemma is see and this a, past tense of that verb or saw is the lemma and it is a noun singular noun. So, this is morphological analysis of the given word.

Then there is a process called tagging. So, where I find out what is the actual, so what is the actual category of this word? So, the difference here is from the morphology analysis that I also have to disambiguate. So, morphological analysis you saw I was giving two different possibilities; in tagging I have to further find out what is the actual correct grammatical category here. So, I have a sentence is Peter saw her; I know the word saw can be either a noun or verb and we also saw their lemmas, but can I say can I tell in this

particular context, the word saw will only be a verb and not a noun. So, can I do the actual disambiguation also; this is that is how tagging is different from the morphological analysis and then we have a morphological segmentation where given a word I can segment to with different morpheme that involved. So, I have a word like denationalization, I said de nation; a l i z ation they are different morphemes that are in this word. So, in general among all these processes, so part of speech tagging is very very popular. So, will words sometime for that and before that we will quickly see what is morphological analysis lemmatization; how it is done?

And finally, there is also these process of generalization where I can take a word, a root word and a particular grammatical category and I have to generate a new word from this. So, see and I want to generate past tense; I want to find out saw in NLP; it is not very popular unless you are talking about national language generation, where you might want to use this.

(Refer Slide Time: 24:39)

What are the applications?

- Text-to-speech synthesis:  
*lead*: verb or noun?  
*read*: present or past?
- Search and information retrieval
- Machine translation, grammar correction

Pawan Goyal (IIT Kharagpur) Computational Morphology Week 3: Lecture 2 15 / 19

Before going into the process, what might be the application for doing this? Why should we be interested in doing morphological analysis? For example, text to speech synthesis, so what is text to speech synthesis you have seen something that is written and you have to find produce the corresponding is speech for that. So, now, if you have a word like lead written somewhere, now depending on whether it is a noun or a verb, you will have a different pronunciation for that lead verses led depending on noun or a verb. So, it is

important of find out from the text whether it is a noun or verb. Same thing goes with read verses red; in general it is very important for other things like search and information retrieval.

So, might want to use the a morphological category to reduce the certain space and also machine translation we saw some example and especially because if you know the morphological category, you can find out the for the target languages what is corresponding affix is or different words that is being used in that languages and grammatical error correction and all that. So, if you know what is the morphological category of this word in the whatever is written, if you can find out this is not the correct morphology that is use you can try to correct it.

(Refer Slide Time: 26:04)

### Morphological Analysis

Input	Morphological Parsed Output
cats	cat +N +PL
cat	cat +N +SG
cities	city +N +PL
geese	goose +N +PL
goose	(goose +N +SG) or (goose +V)
gooses	goose +V +3SG
merging	merge +V +PRES-PART
caught	(catch +V +PAST-PART) or (catch +V +PAST)

**Goal**

To take input forms like those in the first column and produce output forms like those in the second column.  
Output contains stem and additional information; +N for noun, +SG for singular, +PL for plural, +V for verb etc.

Pawan Goyal (IIT Kharagpur)      Computational Morphology      Week 3: Lecture 2      16 / 19

So, now what is morphological analysis, so this is have seen earlier. So, as I input we have words like cats, cat, citizen so on and as an output I want to find out given a word like cats, what is the root verb here cat? What is category noun? And it is a plural cat plus n plus p l something like that. So, in this table, so if I am a given input in the left as per the left column, I want output like the right column. So, the output that I will generate will contain additional information like this is a noun, this is singular for s g for singular p l for plural and v for verb like that. So, this is all the information that I want to get from the given word.

(Refer Slide Time: 26:57)

*Issues involved*

boy → boys  
fly → flys → flies (y → i rule)

Toiling → toil  
Duckling → duckl?

- Getter → get + er
- Doer → do + er
- Beer → be + er?

Now, what might be the issues involved while doing morphological analysis one particular problem is that it is not very very regular. For example, from the word boy I can get plural boys, but what happens if I taken input like fly; I get flies f i l e s. So, this you see that they are following two different sort of tools for doing changes at the bounding. Similarly if I take the word like toiling I can get toil, but what happens if I give an input like duckling, should I used same sort of full to get duckl that is not correct of English word.

So, how do I know the duckling is not a correct English word when an input like a duckling is provides to my system. Similarly from getter I get g e t plus e r from doer d o plus e r, but what happens if given input like beer, do I output like b e plus e r? So, these are some of the issues that involved in processing morphology, now if I have to solve these issues what is what are the different knowledge that I need to have? .

(Refer Slide Time: 28:03)

*Knowledge Required*

*Knowledge of stems or roots*  
Duck is a possible root, not duckl.  
We need a dictionary (lexicon)

*Morphotactics*  
Which class of morphemes follow other classes of morphemes inside the word?  
Ex: plural morpheme follows the noun

*Only some endings go on some words*  
• Do+er: ok  
• Be+er: not so

*Spelling change rules*  
Adjust the surface form using spelling change rules  
• Get + er → getter

Pawan Goyal (IIT Kharagpur)      Computational Morphology      Week 3: Lecture



So, I need to have some knowledge on what are the different words or fruits in English. So, for example, I need to know that duck is a possible root, but duckl is not a possible root in English. So, we need some sort of dictionary or lexicon of English what are different nouns and verbs etcetera in English, what else; I need to have some knowledge of morphotactics what is morphotactics? That is which kind of morphemes follow other kind of morphemes for example, if I have to convert a noun to plural; I know plural morphems always follow the noun. So, this is the morphotactics information which morpheme follows are the morphemes, then I also need to have this information that some endings go only on certain words not on everything.

So, on d o I can apply e r to get doer, but on b e on the verb b e; I cannot apply e r to get a word like beer. So, this is again these constraints also I need to have and then I need to some worry about spell spelling change rules. So, then whenever I have word like get I apply e r it converts to getter. So, there is just duplication on of duplication of the ending t.

(Refer Slide Time: 29:24)

Why can't this be put in a big lexicon?

- English: just 317,477 forms from 90,196 lexical entries, a ratio of 3.5:1
- Sanskrit: 11 million forms from a lexicon of 170,000 entries, a ratio of 64.7:1
- New forms can be created, compounding etc.

One of the most common methods is finite-state-machines

Pawan Goyal (IIT Kharagpur) Computational Morphology Week 3: Lecture 2 19 / 19

So, one question you might have is that why cannot I put everything together in a big lexicon. So, that is all the root words all their morphological variants why cannot they already put in a big lexicon and there I can search even new word, I can search in the lexicon find out what is the actual word and its category if they are finite by country do that. So, there are two different reasons why this may not be very good solution.

So, one is if you take any language like English it is very easy, so where for a given, but they are not many variations in terms of morphology. So, this is some strategies from a data set that in English there are roughly you take 90000 lexical entries and you will find out all the possible morphological forms that you can generate then you find a ratio of 3.521 that is from word you can generate roughly 3.5 more very to variants including that word in English, but this is not true for other languages.

For example, the same thing if you do for Sanskrit, you have a lexicon of 170000 entries, but if you try to derive the forms. So, the ratio that you come up with is something like 64.7 is to 1. So, the 11 million forms that are generate, so these are huge in number. So, 64 is a very big ratio in comparison to 3.5, but again this may for now it may not be a very big number again you can still argue that this might this will it be put in a big lexicon and you can search over that, but there is another problem that is you can keep on coining new words and apply the same morphological processes to generate its forms and you do not know these words a prior. So, you cannot store their morphological

variants in the lexicon, so on the other hand if you can store the what kind of rules in valued for mailing from a word to its plural given a new plural form, you can try to find out its original root word.

So, that is why we will be studying what kind of methods are possibly used and the one of the most popular method in this field of computation morphology. Finally, state methods so this was very very popular earlier first the language like English also and later on for other languages like Indian languages another European language. So, even now if you talk about processing the morphology (Refer Time: 31:54) methods are one of the most popular choices.

So, in this lecture we talked about composition morphology, what are the linguistics terminologies and what is the process as such what is the NLP perspective there and in the next lecture we will talk about how do I use final state methods for this processing.

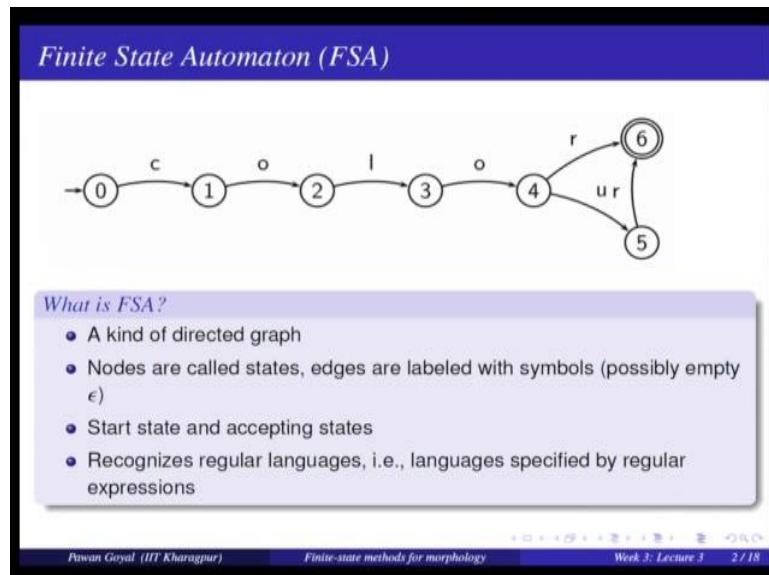
Thank you.

**Natural language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 14**  
**Finite State Methods for Morphology**

So welcome back for the third module of this week. So in the last lecture we were talking about computational morphology. So in this module we will talk about how do we use financial math methods for doing this morphological analysis. So I will just go in briefly into what these models do.

(Refer Slide Time: 00:42)



So starting with what is a finite state automaton. So if you have taken a course on formal languages automaton theory or theory of computation, you might already be aware of what is a finite automaton. I will just briefly tell what is this. If you have not taken a course, that course you might just want to quickly see that in one of the books for that course.

So what is a finite state automaton? So it is kind of a directed graph. So in this figure you are seeing there is a finite automaton that is having 6 different nodes. So these are the nodes here are called the states. And there are edges between the nodes and they are the edges are labeled with certain symbols. So for example, you are seeing between node 0

to 1 there is in there is an edge with the label of c. And they may also be empty if for certain category of this automaton.

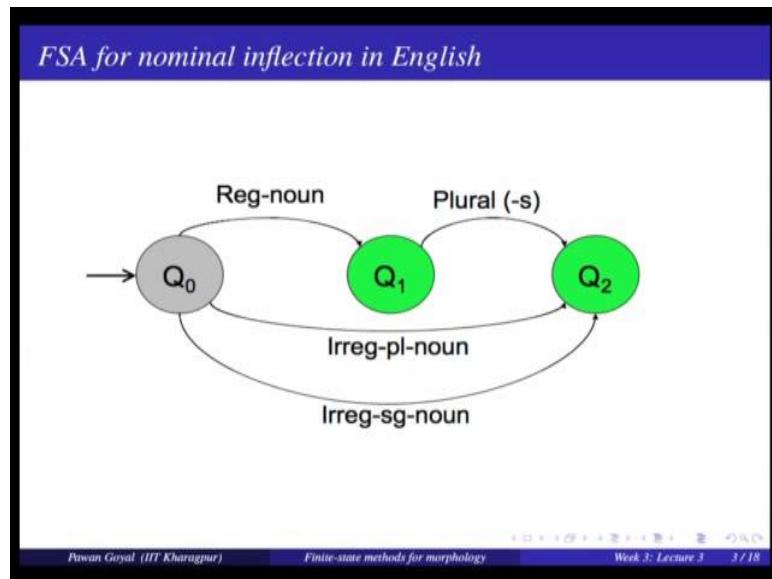
There are certain started states. So for example, here the node 0 is the started state, and there are certain accepting or final states that are noted by maybe having a double circles like on the node 6, you are having double circle this is an accepting state or the finite state. So they can be more than one final state very easily. So now, what do the finite state automaton do? So they recognize a regular language that is the language that is specified by the regular expressions. So any regular expression you can always convert into an automaton financial automaton.

So now if you see the automaton that is provided on this slide, what is the language that it recognizes? What are the words that will be passed through this? And by seeing that you can also see how the automaton actually works. So if you give an input like color to this automaton what will happen? So it will start with the started state, you will take the first character c from a started state on the input character c you will go to the state 1 from this graph. Now state 1 you take an input o that is the next in in the input and you move to state 2 and So on. And as you end with the with the phoneme character r you end up with the state 6 in this graph.

So given an input like see color it is accepted by this automaton, but what happens if I give a word like colour you can see that there is still a path. You can go from 4 to 5 and 5 to 6 and this automaton you can accept colour. So it will accept 2 words color colour, but what happens if you give it is some other word like colr. So you will see col will go to state 3, but if you take r there is no path from 3 that takes an input r. So it cannot move further.

So this input will not be accepted by this automaton. So given an automaton final state automaton what it does? It recognizes if a finite, so not finite a regular language. So if you give any string from the regular language it will accept; that means, this will end up in one of the accepting states, but you should give it any other string that is not in the language it will not accept. So this will not end up in any of the accepting states.

(Refer Slide Time: 04:18)

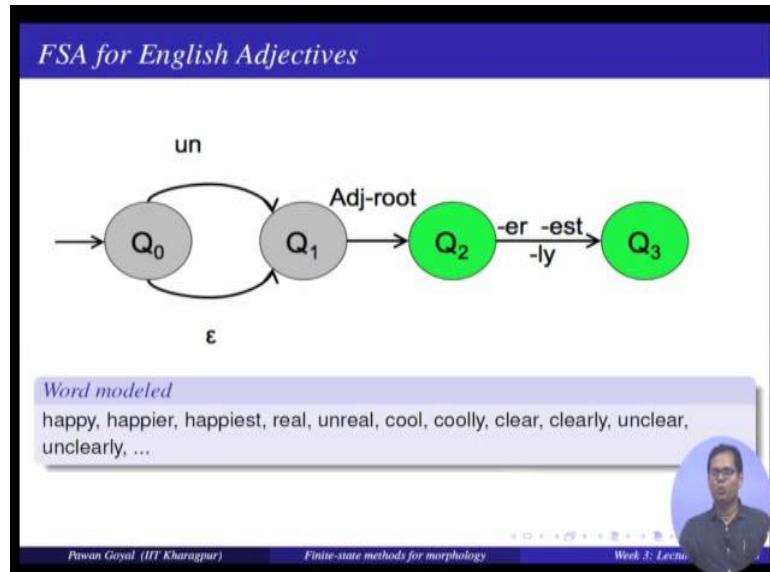


Now how are this finite state automaton used for doing the morphological analysis? So what is the idea? The idea is that when you combine various morphemes there are certain changes that happen at the boundary mainly concatenation. And this is very regular phenomenon. So if you want to capture that 2 words like boy and car can be made converted into plural by adding an s, I can simply have a state for the noun where boy and car both come together and then I have a single edge from there with s marking that there is a plural. And now you think about all the possible plural that you can make in English. All the singular nouns can come to the same state and then you have a single arrow that with s and that will convert into fluid. So that very efficiently captures the process of converting in from singular to plural and as long as this is regular this is very straightforward then how do you do that.

So here is an example in this slide. So only 3 states are shown here Q naught, Q 1, Q 2 - Q naught the starter state and Q 1 and Q 2 other accepting or finite states. Many other intermediate states are not shown we will see that in some of the later slides. So from Q naught to Q 1 you have the regular nouns like car boy bag etcetera and then so all of these are also words in English. You can now add a plural morpheme s to convert them into plural. So going from Q 1 to Q 2, but as such what happens if you have an irregular noun like goose to geese. So that cannot go to Q 1 because to goose if you apply as you will not get its plural form. So for the irregular nouns you have a different path that goes from Q naught that it to Q 2 again there will be some intermediate nodes

for individual characters of phonemes in in that in the language in the world. So both the singular plural will go to Q. So Q 1 and Q 2 both will accept the words in the English.

(Refer Slide Time: 06:35)



Similarly, this is in FSA finite state automaton for English adjectives. So you have some prefix like un then you have the actual adjective root like happy and then you have certain suffixes like er, est, ly. So you see here we are also showing some lot of morpho tactics that what kind of morphemes follows other kind of morphemes. So now, what kind of words that you can generate by using this automaton? So you can say un happy - unhappy. A starting from Q naught or from Q naught you taking absolute transition go to Q 1 and you have happy and er – happier, happiest, happily and so on. So sick is generate words like happy, happier, happiest real from Q naught you take an absolute transition go to Q 1 then you have the adjective root real that can Q 2 is a final state. So you can generate real. If you have generated unreal from Q naught you take un that will give you un plus have the word real and go to Q 2 you become, you get unreal. Similarly, all other words you can generate by using this particular automaton.

(Refer Slide Time: 07:49)

*Morphotactics*

- The last two examples model some parts of the English morphotactics
- But what about the information about regular and irregular roots?

*Lexicon*

Can we include the lexicon in the FSA?

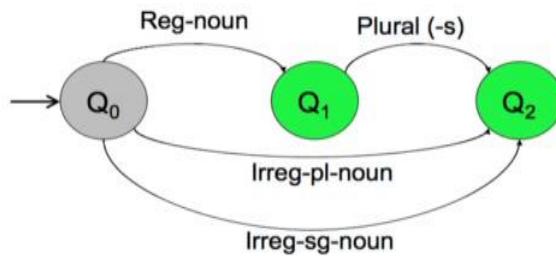


Pawan Goyal (IIT Kharagpur)    Finite-state methods for morphology    Week 3: Lecture 2

So as we already said in these examples we also seen some English morphotactics, that is what kind of morphemes come after another kind of morphemes in English. Now but what do I do about regular and irregular roots in English, how do I capture that information in this automaton? So can we include the lexicon also in my automaton? So that is what we will see. If I also want to include words like car, cars, bag everything inside my automaton, how do I do that?

(Refer Slide Time: 08:29)

*FSA for nominal inflection in English*



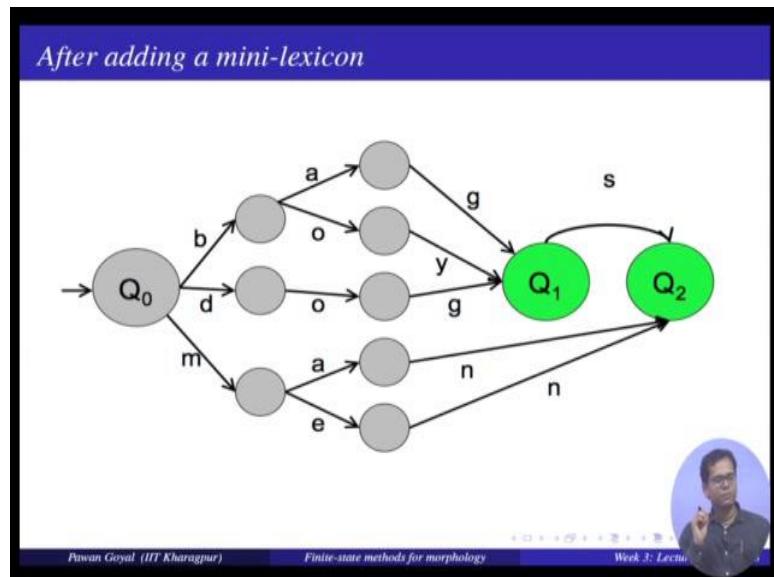
```
graph LR; Q0((Q0)) -- "Reg-noun" --> Q1((Q1)); Q0 -- "Irreg-pl-noun" --> Q2((Q2)); Q1 -- "Plural (-s)" --> Q2; Q2 -- "Irreg-sg-noun" --> Q0;
```



Pawan Goyal (IIT Kharagpur)    Finite-state methods for morphology    Week 3: Lecture 2

So this is what we have seen. So from Q naught I take all the regular nouns to Q 1 all the irregular nouns directly go to Q 2, but now I want to include the lexicon. So let us take a very simple and a small lexicon.

(Refer Slide Time: 08:42)



So I have the words like bag boy and dog as the regular noun and man as the irregular noun. And we want to generate all the, so I want to recognize all the singular and plural form. So what do I do? So you see here from the state Q naught I am now having all the possible other individual states that take the regular singular nouns to Q 1. So I have different nodes for b and then the b shared among bag and boy. Similarly, there is a different state for going to doc. And finally, all of these they go to Q 1 all the regular nouns in the singular form and then you get you can have s added to that this becomes plural.

So you have dogs, bags and boys together. What do you do for the irregular nouns you have a different path? Man and m a n and m e n they again go to Q 2 directly. Now given this figure can you recognize the words like boys easily, yes I start with Q naught when I get the word b I go to the next state o, I go to another state y I go to Q 1 and s I go to Q 2. So I can recognize boys I can recognize boy m a n, m e n all this can be recognized by this automaton. I can further expand it to include other words my vocabulary. But the education here is this; the goal of morphological asks that we started with. So what is this automaton doing? Given a word in English it will tell me whether it is a singular or

plural word in English. Assuming that, you have taken care of all the words in your vocabulary while building this automaton, so it can recognize various words.

(Refer Slide Time: 10:36)

The slide has a blue header bar with the text "Some properties of FSAs: Elegance". The main content area contains a bulleted list of properties:

- Recognizing problem can be solved in linear time (independent of the size of the automaton)
- There is an algorithm to transform each automaton into a unique equivalent automaton with the least number of states
- An FSA is deterministic iff it has no empty ( $\epsilon$ ) transition and for each state and each symbol, there is at most one applicable transition
- Every non-deterministic automaton can be transformed into a deterministic one

At the bottom of the slide, there is a video player interface showing a thumbnail of a person, the name "Pawan Goyal (IIT Kharagpur)", the course title "Finite-state methods for morphology", and the lecture number "Week 3: Lecture 1".

So that is what we have written here. So what are the properties of FSAs? So they are very elegant. So that is the recognition problem can be solved in linear time. What do I mean by that? Given give me any input string. So you can find out whether automaton recognizes the string or not a linear time linear in the length of the string. Because every time you are making if you are having a word like boys you are checking if from the start state if you go, if you take input b where do you go o y s and then finally, if you end up in the accepted state, you will accept the string otherwise you will not, of course, this will happen only for the deterministic automaton, but as in the elegance of itself we know, that every non-deterministic finite automaton can be converted to a deterministic finite automaton and there is a simple algorithm for that.

So we do not have to worry about it. Even if we have started building an NFA I non-deterministic automaton I can convert it to DFA. Where the linear time you can find out whether a string is accepted or not. Similarly, I do not have to worry about getting the minimum mode of states of the automator, because the again there is an algorithm that converts any automaton into it the equivalent automaton that has the minimum number of states.

So this is why we say this is very elegant you can convert an NFA to DFA. And in DFA you can convert it into the minimum number of states since very elegant. And we have seen that it can work as a language recognizer given a regular language it can tell me a new string whether this is in the language or not, but. So coming to my previous question is it what we need in the morphological analysis. So answer is no, this is not sufficient for the morphological analysis we, so take the word boys. Remember what are the different morphology analysis we talked about, we talked about finding the lambdaization, is like the simplest one lambdaization. I want to find out what is the lemma for the word boys. I want to find out the word boy, can the DFA help me to obtain the word boy? It cannot, it can only tell me that boys are a word in me in my defined regular language, this is there. So this is either single plural, but it cannot tell whether boys is what is the lemma for the word boys. So I need some other more model that can give a word can also give me what is the lemma or the root form.

(Refer Slide Time: 13:28)

*But ...*

FSAs are language recognizers/generators.  
We need transducers to build Morphological Analyzers

*Finite State Transducers*

- Translate strings from one language to strings in another language
- Like FSA, but each edge is associated with two strings

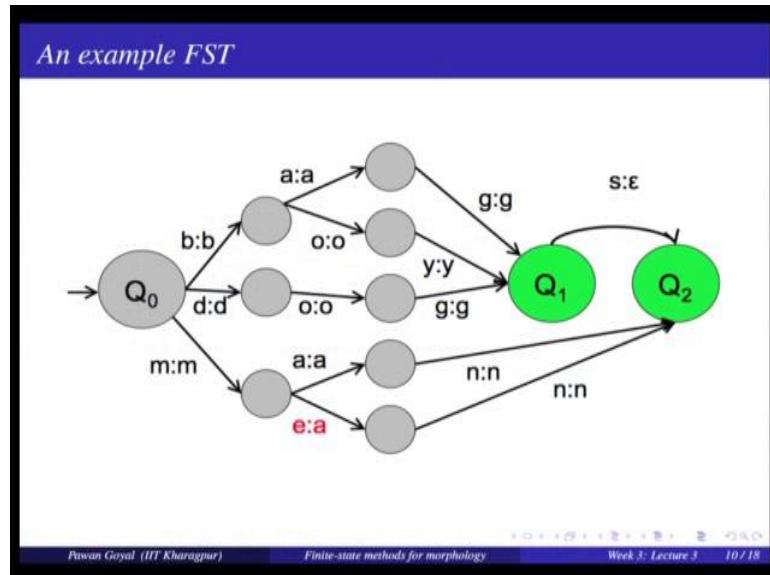
Pawan Goyal (IIT Kharagpur)      Finite-state methods for morphology      Week 3: Lecture 3      9 / 18

So for that, So FSAs are language recognizer or generators. So we need instead transducers that can help me do morphological analysis.

So what are transducers? So transducers are very similar to final state automaton. Except that they can help me translate one string into another. So now, what happens there? How do they do that? The model is very similar to FSA, but now in each edge of the FSA instead of having a single phoneme as the label I have the input phoneme or input

characters' symbol and the output character symbol. So it translates one-character symbol to another character symbol. So that solves my problem of going from the input string like boys to an output this string like boy.

(Refer Slide Time: 14:23)



So let us take the example on the same lexicon of forwards and they are plural. So earlier we saw the final decision automaton where we were having each edge label within input character only. So finally, it could recognize whether a word is there in the language or not, but now we have the transducer where each other is labeled with the input as well as the output character. So you see here. So here m, m e n and e is the input a is the out. So once you give an input like m e n to this transducer the output you will get is m e n the actual lemma. That is how they can solve the morphological analysis problem.

(Refer Slide Time: 15:11)

Given the input *cats*, we would like to output *cat+N+PL*, telling us that cat is a plural noun.

We do this via a version of **two-level morphology**, a correspondence between a lexical level (morphemes and features) to a surface level (actual spelling).

Lexical	{	c	a	t	+N	+PL	}
Surface	{	c	a	t	s		}

Pawan Goyal (IIT Kharagpur)      Finite-state methods for morphology      Week 3: Lecture

Now, some small details that what might be the problem that that you might face while doing that. So for example, here is an input. So we are calling it the lexical label, cat I want to convert it to its noun, but a plural form. So I know cat gives me cats, but what happens if I take a word like fox. Fox will give me foxes. So there will be they are changed at the boundary you adding an e between fox and s. So now, there are many ways in which you can handle it. One is you have you take cat has a regular noun and then have a regular plural to that and fox as irregular noun separately this is one possibility.

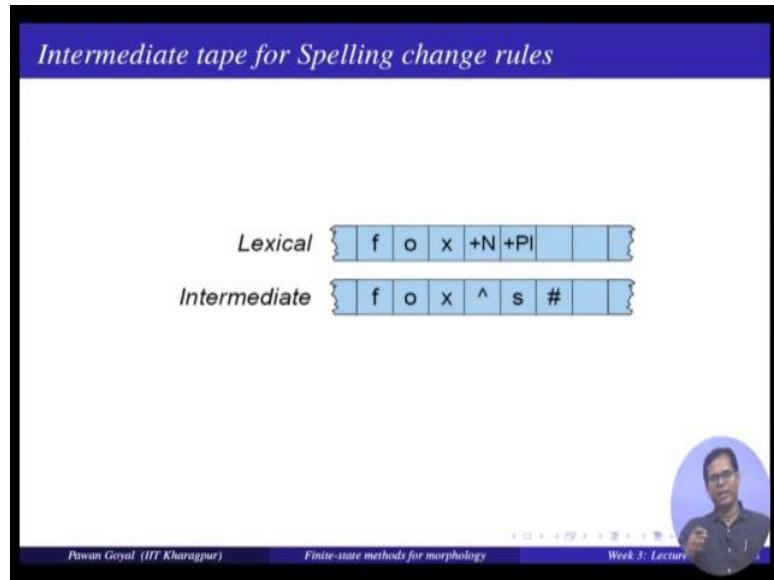
But the interesting idea is if you can use a 2 level morphology that is between the lexical form cat plus noun plus plural and the surface form, that is cats can I define an intermediate form. That is cat there is an s, but in between there is some placeholder that can be null or e depending on what are the phonemes available in the stem and the surface. So that is instead of going from lexical to surface level directly can I go to the individual level, first this is my first label and then I transfer from intermediate label to the surface level in the second level of morphology. So this is my 2 level morphologic.

(Refer Slide Time: 16:50)

*Intermediate tape for Spelling change rules*

Lexical    { f o x +N +Pl }

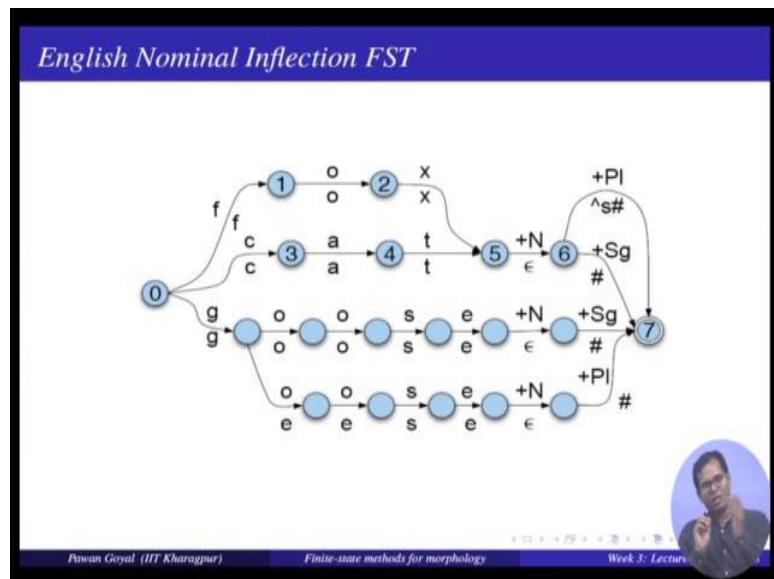
Intermediate { f o x ^ s # }



Pawan Goyal (IIT Kharagpur)      Finite-state methods for morphology      Week 3: Lecture

So example is, so the motivation example is for cats and fox. So instead of going from fox plus n plus p l to foxes I go to fox and some placeholder and an s and this is the end of word. Now I can have a second label rule that says if my (Refer Time: 17:06) fox of x is h, s will there be some addition in the in between will there will be sudden change in between that can be taken care of by a separate module. Similarly cat there is a placeholder an s I can find out what should be the actual surface from given this intermediate form.

(Refer Slide Time: 17:26)



So here is an example. So I have in my vocabulary words like fox, cat, goose and I am generating the singular and plural form. So what are you seeing here? From fox to generate the plural form same with cat I add a placeholder as and the end of the character. So that is here now. So this is my intermediate form so; that means, at this point I am not distinguishing between fox and cat they are behaving very similarly. So I need an additional process to find out given this intermediate form what should be the surface level form.

(Refer Slide Time: 18:23)

*Spelling Handling*

A spelling change rule would insert an e only in the appropriate environment.

<i>Lexical</i>	f	o	x	+N	+Pl	
<i>Intermediate</i>	f	o	x	^	s	#
<i>Surface</i>	f	o	x	e	s	



Pawan Goyal (IIT Kharagpur)      Finite-state methods for morphology      Week 3: Lecture

So this is 2 level morphology. I go to fox s there is something in between that may be null or something else intermediate label and then from intermediate labeled you go to the surface level.

Now, what do you think the transition depend depends on from intermediate or surface label? So it will depend on what are the ending characters of the stem and the starting director of the affix, now given these, what should be the replacement character.

(Refer Slide Time: 18:56)

Rule Notation  
 $a \rightarrow b/c_d$  : "rewrite a as b when it occurs between c and d."

Pawan Goyal (IIT Kharagpur)      Finite-state methods for morphology      Week 3: Lecture 1

So you can have these simple rules in this notation. This is also called Catherine cook k notation for condensed context as sensitive rules. So that so rules are that a letter a h converted to b if h visited by c and followed by d. So this is if you remember the context sensitive grammar.

(Refer Slide Time: 19:24)

$a \rightarrow b/c_d$

c a d → c b d

Inter. → Surface

So that is what I am saying h converted to b if preceded by c and forward by d; that means, whenever a comes between c and d, then you convert a to b. This is the context

sensitive rules. So this kind of spelling change rules you can apply to go from intermediate label to the surface level you can use this rules to do the conversion.

(Refer Slide Time: 20:00)

The slide has a blue header bar with the title 'Morphological Analysis: Approaches'. Below the header is a light purple box containing the text 'Two different ways to address phonological/graphemic variations'. Inside this box are two bullet points:

- Linguistic approach: A phonological component accompanying the simple concatenative process of attaching an ending
- Engineering approach: Phonological changes and irregularities are factored into endings and a higher number of paradigms

At the bottom of the slide, there is a navigation bar with icons for back, forward, search, and other presentation controls. The footer contains the text 'Pawan Goyal (IIT Kharagpur)', 'Finite-state methods for morphology', 'Week 3: Lecture 3', and '16 / 18'.

Now so after talking about this 2 level morphology, in general when people design the morphological analysis for a given language, they follow 2 different kinds of approaches. So they are also called the linguistic approach and engineering approach.

So what is the difference between the two? So in the linguistic approach what will happen? You have the stem, stem is already defined in the lexicon or the language and you have an affix that is also defined. So to the stem you will apply the suffixes h, but the surface changing rules will be taking care of separately by the rules of this kind, that we have just seen. So there will be rules of this kind that will define the surface how, what are the changes that are happening at the surface level.

On the other hand, engineering approaches, you would not have the separate rules for spelling changes. They will try to find out the minimum possible unit of the stem that can be used and to that what other and the affixes can be big and in that case that can be applied to the stem. Let us see one example that makes it clear. So the idea is that engineering approach all the phonetic irregularities will be factored into the endings.

(Refer Slide Time: 21:24)

### Different Approaches: Example from Czech

	woman	owl	draft	iceberg	vapor	fly
S1	zen-a	sov-a	skic-a	kr-a	pár-a	mouch-a
S2	zen-y	sov-y	skic-i	kr-y	pár-y	mouch-y
S3	zen-e	sov-e	skic-e	kr-e	pár-e	mouch-e
P2	zen-0	sov-0	skic-0	ker-0	par-0	much-0

A linguistic approach

$$\begin{array}{llllll} \text{zen} + \left\{ \begin{array}{l} \text{a} \\ \text{y} \\ \text{e} \\ \text{0} \end{array} \right. & \text{sov} + \left\{ \begin{array}{l} \text{a} \\ \text{y} \\ \text{e} \\ \text{0} \end{array} \right. & \text{skic} + \left\{ \begin{array}{l} \text{a} \\ \text{y} \\ \text{e} \\ \text{0} \end{array} \right. & \text{kr} + \left\{ \begin{array}{l} \text{a} \\ \text{y} \\ \text{e} \\ \text{0} \end{array} \right. & \text{pár} + \left\{ \begin{array}{l} \text{a} \\ \text{y} \\ \text{e} \\ \text{0} \end{array} \right. & \text{mouch} + \left\{ \begin{array}{l} \text{a} \\ \text{y} \\ \text{e} \\ \text{0} \end{array} \right. \end{array}$$

An engineering approach

$$\begin{array}{llllll} \text{zen} + \left\{ \begin{array}{l} \text{a} \\ \text{y} \\ \text{e} \\ \text{0} \end{array} \right. & \text{sov} + \left\{ \begin{array}{l} \text{a} \\ \text{y} \\ \text{e} \\ \text{0} \end{array} \right. & \text{skic} + \left\{ \begin{array}{l} \text{a} \\ \text{i} \\ \text{e} \\ \text{0} \end{array} \right. & \text{k} + \left\{ \begin{array}{l} \text{ra} \\ \text{ry} \\ \text{fe} \\ \text{er} \end{array} \right. & \text{p} + \left\{ \begin{array}{l} \text{ára} \\ \text{áry} \\ \text{áfe} \\ \text{ar} \end{array} \right. & \text{m} + \left\{ \begin{array}{l} \text{oucha} \\ \text{ouchy} \\ \text{oute} \\ \text{uch} \end{array} \right. \end{array}$$


Pawan Goyal (IIT Kharagpur)      Finite-state methods for morphology      Week 3: Lecture

So here is one example from check. So what you are seeing here. So they are various words women, owl, draft, iceberg, wrapper and fly. For each of these you have the actual word and s that is like for women, zen z e n and owl it is sov. So here the root word and there are various affixes are applied in various grammatical functions that are abstracted using S1, S2, S3 so on and P2, the various grammatical functions. You might treat them as singular plural and so on.

So to a given root word where is affixes have been applied. So what you are seeing here are the final forms. So for the initial 2 words they seen pretty regular. So you have the word sov you apply a y e n o for getting different forms similarly, for zen, skic you see some difference right they are in red. So what is the difference that you are seen? So now, the suffix why has been changed to I in the case of S2. Similarly, the accent is gone over e from me and you are getting the single simple e. So that is why it is in red. So this is an irregularity. This is okay, but what happens if we go to ice work with kr. You see in the case of S3 even the stem that is a change. So on r you are getting an accent. And for P2 you are getting another word another character e in between kr. So you are getting k e r. So now so on you will see such changes even vapor and fly. So now, the question is how does one capture all this - all the changes, all these irregularities in the case of stems and the suffixes.

So now in the linguistic approach what will happen, I will take the same route word. So for example, here I will have skic as the root word and kr as the root word. And the same set of suffixes. So y a y e with an accent you know are my suffixes. That is my root and the suffix pair I will add these and whatever a spelling change rules I will try to enumerate, whatever different spelling change rules, by taking what is the previous character previous to previous character, I will try to define the spelling changes in that way. That is a linguistic approach.

What will happen in engineering approach? I try to find out the part of the stem that is common across all these variations. So it is with the first 3, zen, sov and skic, but with kr, I will take only k as common, I will not take r because r is getting changed into something with an accent or with an e n r. So I will take only k as common is part of my stem, and everything else remaining I will convert into my stem that I might add. And that you see in the last three. So you have only k p n m as the actual strength and everything else goes into the suffix. So this is what is engineering approach. So by doing that you do not have to worry about handling the spelling change rules separately you are you handling them here itself.

(Refer Slide Time: 24:55)

The screenshot shows a presentation slide with a blue header bar containing the text "Tools Available". Below the header, there is a list of two items:

- AT&T FSM Library and Lextools  
<http://www2.research.att.com/~fsmtools/fsm/>
- OpenFST (Google and NYU)  
<http://www.openfst.org/>

At the bottom of the slide, there is a small video player window showing a man speaking, identified as Pawan Goyal from IIT Kharagpur. The video player has a circular progress bar and some control icons. The footer of the slide includes the text "Pawan Goyal (IIT Kharagpur)", "Finite-state methods for morphology", and "Week 3: Lecture".

So there are various tool kits set available for doing the morphological pricing, for example, AT and FSM library that is very popular. Again the OPENFST tool is very popular for doing the morphological analysis for a given language. So this is this was

about our computation morphology using finite state methods. So I have very briefly talked about what our financial methods and how do we how do you use that for computational morphology.

So in the next lecture we will talk about in the same process a more popular problem about part of speech tagging. So what is the problem part of speech tagging and what are the different computational models that you can use to handle that.

Thank you.

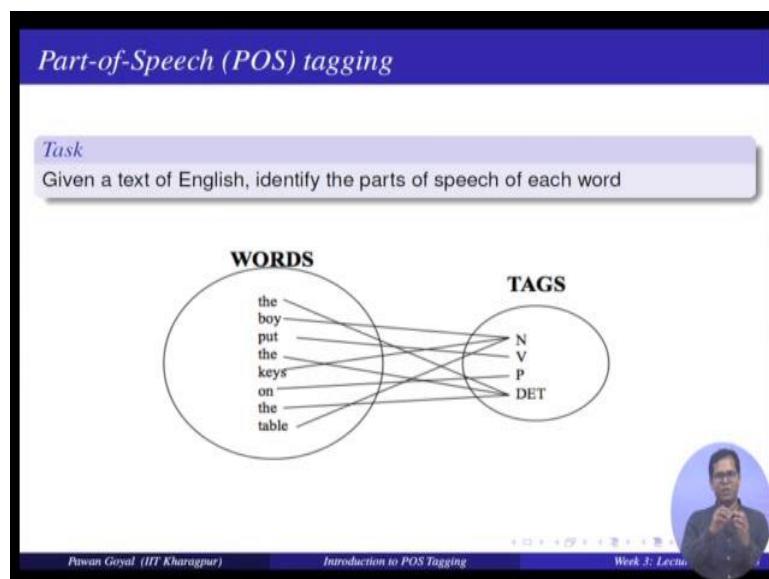
**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 15**  
**Introduction to POS Tagging**

Welcome to the 4th module of this week. So as we said last time, we will be starting with a very very popular problem of part of speech tagging, if you remember this was one of the processes that we talked about when dealing with morphology. So, you can do lamentation, you can do morphological analysis, you can also tagging. So, what was the thing that was different in tagging that you are not doing morphological analysis, if you remember, you will also finding out among all the possibilities what is the actual particular morphological degree that this 4 things.

You also are doing the disambiguation here. So, this is my problem of part of speech tagging that given a set of words so; that means, a sentence or document whatever can I identify what is the actual category for each of the word, I have to give the unique answer for each one. So, if we take example for English text. So, given a text of English can be sentence or whatever can identify the part of speech of each and every word that is the problem of part of speech tagging.

(Refer Slide Time: 01:17)



If I have a sentence like the boy put the keys on the table and I have 4 possible part of speech tags. So, very very coarse label like noun and pronoun v for verb and p for pronoun the first p here is for prepositions and d DET for the determiner. So, you can map – the, is a determiner and boy, is a noun; put, is a verb; the, is a determiner; keys, a noun and on, is a preposition and the, is a determiner and table, is a noun.

(Refer Slide Time: 02:19)

*Parts of Speech: How many?*

*Open class words (content words)*

- nouns, verbs, adjectives, adverbs
- mostly content-bearing: they refer to objects, actions, and features in the world
- *open class*, since new words are added all the time

*Closed class words*

- pronouns, determiners, prepositions, connectives, ...
- there is a limited number of these
- *mostly functional*: to tie the concepts of a sentence together

Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture

Each word you can map to one of the 4 tags. So, this is a problem that given a sentence, you have all the individual words can identify and define what is in grammatical category for each of this word.

Now, so, one natural question that you might have is so, how many different part of speech tags are there? Firstly, let us see, what are the different things that will go in the part of speech part of speech tag categories? Firstly, you will have the open class words that are sort of content words. So, they are like nouns, verbs, adverbs, adjectives they are there, they will have their own part of speech text. So, they are the open class words are those that we are the content mostly and, so why do we call them open class because you keep on adding new and new words in the language we have coin we have seen a term for this.

You have words like Google, Photoshop, Skype; they come into the language all. So, these are coming into nouns or verbs in the language. So, this is open class you can keep on adding new and new words and then you have the closed class words like pronouns

here very fixed set of pronouns determiners very a fixed then prepositions connectives and all that they are very fixed they are all functional words and they are closed class and as you know they are they are used to tie various concepts the sentence together.

I need some part of speech categories for open class words some categories for closed class words. So, one possibility can be I can choose very very coarse grained categories.

(Refer Slide Time: 03:53)

*POS examples*

■ N	noun	chair, bandwidth, pacing
■ V	verb	study, debate, munch
■ ADJ	adj	purple, tall, ridiculous
■ ADV	adverb	unfortunately, slowly,
■ P	preposition	of, by, to
■ PRO	pronoun	I, me, mine
■ DET	determiner	the, a, that, those



Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture

Like here, so I can take noun so I am also giving some examples here like chair bandwidth pacing all these are nouns, I can take verb, study, debate, munch, all these are verbs then adjective like purple tall ridiculous they are adjectives adverbs unfortunately slowly proportion of by, to, they become my preposition pronoun I, me, mine and then determiner, the, a, that, those.

Maybe I can just use it very very coarse label part of speech tags. But think of the perspective when of language pronouncing, so what is helpful to me? To take a very coarse label or to go to the more fine grained label. So, here what will happen? I can find out which word is a noun, but if I tell it is a singular noun a plural noun I cannot tell if I have even if I have the tagging information. So, I might want to prefer a tagging scheme where I also go to some sort of grammatical details of the word, it is a noun and is it a verb is it a the third same person singular verb is it a verb plus a past tense and so on even noun, is it a proper noun?

I might want to go into finer detail, but again I cannot go into very very finer details. So, otherwise there will be too many part of speech tags.

(Refer Slide Time: 05:21)

*POS tagging: Choosing a tagset*

- To do POS tagging, a standard set needs to be chosen
- Could pick very coarse tagsets  
*N, V, Adj, Adv*
- More commonly used set is finer grained, "UPenn TreeBank tagset", 45 tags

*A Nice Tutorial on POS tags*  
<https://sites.google.com/site/partofspeechhelp/>

Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture 1

There are various schemes that are available, they have their own part of speech tags definitions, but given a sentence if you have to do part of speech tagging, you need to first define what is your tagset. What is all the possible categories among which you have to choose it and there are various standards available.

I can choose very very coarse tagsets like only taking noun verb adjective adverb or I can take a good enough set that gives me some additional grammatical information with each of the word each of the word. So, one for very popular part of speech tagset is university Pennsylvania, UPenn, Treebank, tagset that contains every 45 part of speech tags and we will see examples also and there is a very nice tutorial on these part of speech tags in terms of what is the difference between this part of speech tag versus at part of speech tag, particle versus adverb how are they different by giving examples. So, that I recommend as if you want to get more information in the sense of how they are used in various sentences you can look at this site.

(Refer Slide Time: 06:34)

UPenn TreeBank POS tag set					
Tag	Description	Example	Tag	Description	Example
CC	Coordin. Conjunction	and, but, or	SYM	Symbol	+, %, &
CD	Cardinal number	one, two, three	TO	"to"	to
DT	Determiner	a, the	UH	Interjection	ah, oops
EX	Existential 'there'	there	VB	Verb, base form	eat
FW	Foreign word	mea culpa	VBD	Verb, past tense	ate
IN	Preposition/sub-conj	of, in, by	VBG	Verb, gerund	eating
JJ	Adjective	yellow	VBN	Verb, past participle	eaten
JJR	Adj., comparative	bigger	VBP	Verb, non-3sg pres	eat
JJS	Adj., superlative	wildest	VBZ	Verb, 3sg pres	eats
LS	List item marker	1, 2, One	WDT	Wh-determiner	which, that
MD	Modal	can, should	WP	Wh-pronoun	what, who
NN	Noun, sing. or mass	llama	WPS	Possessive wh-	whose
NNS	Noun, plural	llamas	WRB	Wh-adverb	how, where
NNP	Proper noun, singular	IBM	\$	Dollar sign	\$
NNPS	Proper noun, plural	Carolinas	#	Pound sign	#
PDT	Pradeterminer	all, both	"	Left quote	(" or "')
POS	Possessive ending	's	"	Right quote	(' or ")
PRP	Personal pronoun	I, you, he	(	Left parenthesis	(, [, {, <
PRPS	Possessive pronoun	your, one's	)	Right parenthesis	(,), }, >)
RB	Adverb	quickly, never	,	Comma	,
RBR	Adverb, comparative	faster	:	Sentence-final punc	(, ! ?)
RBS	Adverb, superlative	fastest	:	Mid-sentence punc	(: ; ... - -)
RP	Particle	up, off			

Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture 4      6 / 18

Here are some examples of the Treebank, the UPenn Treebank part of speech tag. So, what are the part of speech tags they used and what are the various sort some example words here. So, for example, if I see conjunction so, and, but, are, all the conjunction they give a tag like CC, 1, 2, 3, they are cardinal numbers, they give a tag like CD a, the, they are determiner - give a tag of DT, then additional there foreign word then, off, in, by - they are preposition, yellow - adjective, then bigger so, again with adjective you have comparative, superlative. So, you are adapting some more information there (Refer Time: 07:16).

Now, if you see at the nouns, you have a tag NN for only the noun, it is a singular noun, a masculine, sorry, it is a singular noun, for plural noun you have a tag like NNS, but, now you see there an extra text, for proper noun you had separate tags. So, once you know the tag you know it is a noun or it is a proper noun and whether it is a singular or plural you get all this information. So, that is why getting it into some finer details is helpful. So, you have 4 different texts for nouns.

Similarly, so if I am escaping those I am going to the verb directly. So, here text like VB for the base form of the verb then VBD for the past tense, VBG for gerent, VBN for past participle and then third singular present is VBP and third singular and is like VBZ non third singular versus first singular there are 2 different forms. So, you have different

forms of verbs, again that they give you various grammatical information and then you can see other sort of part of speech tag that are available in to UPenn trees tagset.

(Refer Slide Time: 08:34)

*Using the UPenn tagset*

*Example Sentence*

The grand jury commented on a number of other topics.

*POS tagged sentence*

The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture

Let us take an example then. So, we have taken a taking a sentence and if we tag it by UPenn tagset, how does it look like. So, the sentence is the grand jury commented on a number of other topics. So, now, here the word does determiner grand jury adjective. So, jury would be a noun commented becomes a verb - you have seen the part of speech tags, on is a preposition i n a is determiner number is a noun of is again a preposition other adjective and topics they come in noun in plural form. So, this is the information that you get by the part of speech tags.

(Refer Slide Time: 09:27)

*Why is POS tagging hard?*

*Words often have more than one POS: back*

- The back door: *back/JJ*
- On my back: *back/NN*
- Win the voters back: *back/RB*
- Promised to back the bill: *back/VB*

*POS tagging problem*

To determine the POS tag for a particular instance of a word

Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture 1

Now, the question might come that is why are we worried about solving this problem of part of speech tagging, it is a hard problem or a trivial problem. Can I define for each word in my lexicon what will be its part of speech tag? It is a very very simple problem no. So the problem occurs because each word does not have a unique part of speech tags and it might depend on the context in which the word is being used. So what I am saying is that a word might have multiple parts of speech tags and only by seeing the context you might be able to identify what is the actual part of speech tag is being used in this particular context.

Let us take an example. So, I have the word like *back*. So, what are the parts of speech tags that this simple word like *back* can help. So, if I take the sentence like *the back door* what is *back* here, what is the part of speech tag of *back*? So, it is an adjective now if I take a word like, sentence like *on my back* it is not an adjective anymore it becomes a noun in this case yes. Now I have it *win the voters back* of the sentence it becomes *in win back*, so it becomes an adjective and if I have a sentence *promised to back the bill* now this becomes a verb. So, the same word *back* can be used in multiple parts of speech tags.

Immediately you can see the problem that given the context find out what is the appropriate part of speech tag to be used. So, to determine the part of speech tag for a particular instance of a word is my part of speech tagging problem.

(Refer Slide Time: 11:08)

*Ambiguous word types in the Brown Corpus*

*Ambiguity in the Brown corpus*

- 40% of word tokens are ambiguous
- 12% of word types are ambiguous
- Breakdown of ambiguous word types:

<b>Unambiguous (1 tag)</b>	35,340
<b>Ambiguous (2–7 tags)</b>	4,100
2 tags	3,760
3 tags	264
4 tags	61
5 tags	12
6 tags	2
7 tags	1 ("still")

Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture 1



Now, what are the various information that we can use for doing this, for how common is this first of all how common is this problem how many words are actually ambiguous in terms of part of speech tags. So, if I see if I want to see that from the data. So this is your this, so, this point I just want to say something like once you encounter problem in language or any other field, the first thing that you might want to see is that how common is that problem if it is a very very rare problem maybe it is not worth to spend too much time you can have simple rules for solving that, but it says if it is a very very common problem then yes you might have to tackle it using certain models.

Similarly here the disambiguation part is speech tags. So, then it happens only for 10 words if it happens for only for 10 words, I need not worry about building some models and all that for solving this problem, but if it happens for say 10 percent of the words then yes there is a real problem and I need to think of some model for solving it. So, now, if let us see from the corpus how common is this ambiguity problem. So, if I take the brown corpus, what we see here 40 percent of the word tokens are ambiguous and 12 percent of the word types ambiguous.

I hope you remember the distinction between types and tokens are the all the passes all the occurrences of different words. So, same word occurs multiple times are multiple tokens. So, 40 percent of word tokens are ambiguous. So, what we are saying? Saying is that in a corpus if I am encountering tokens 40 percent of them are ambiguous that is

a huge number 40 percent of all the words that occurred in the corpus ambiguous so; that means the real problem.

Now, if we want to just break down of the ambiguity type that how many unique words have different number of tags. So, what we see here. So, roughly 35000 types have only 1 part of speech tag, now 3760 types have 2 tags and 264 have 3 tags and so on and there is 1 word like is still that has got 7 different part of speech tags in the brown corpus. So, yes, getting 6 to 7 tag is very very rare, but getting 2 and 3 tags is quite common especially 2 tags is very very common in the brown corpus.

(Refer Slide Time: 13:36)

The slide has a purple header bar with the text "How bad is the ambiguity problem?". The main content area contains the following bullet points:

- One tag is usually more likely than the others.  
In the Brown corpus, *race* is a noun 98% of the time, and a verb 2% of the time
- A tagger for English that simply chooses the most likely tag for each word can achieve good performance
- Any new approach should be compared against the unigram baseline  
(assigning each token to its most likely tag)

At the bottom of the slide, there is a video player interface showing a thumbnail of a person, the name "Pawan Goyal (IIT Kharagpur)", the title "Introduction to POS Tagging", and the text "Week 3: Lecture".

Now the next problem so we have solve that this problem is frequent yes this is not a real problem, but how bad is this problem what do I mean by this. So, can we always identify for a given word is one tag more likely than another. So, let us take an example of the word race - race can be a noun and a verb when the brown corpus race at race occurs as a noun 98 percent of the time, but as a verb 2 percent of the time.

If I have a simple model that always assigns the word race to a noun that will immediately achieve a 98 percent accuracy for this particular example at least; that means, whenever I am trying to design a model I should be able to think of what is simple baseline uninitialized compare if I am making computational model that is working even worse than this baseline it may not be very very helpful. So, despite this

can be my simple baseline for testing any of the model that I will propose for this particular task.

A tagger for English that simply chooses the most likely tag for each word can achieve good performance. So, it can be even more than 90 percent. So always it is not good to look only at the final numbers, it is also good to see how much you are improving over maybe some of the simplest baselines and some other models that are there in the literature. So, at least the simple baseline how much you are able to do better than the simple baseline.

(Refer Slide Time: 15:25)

*Deciding the correct POS*

*Can be difficult even for people*

- Mrs./NNP Shaefer/NNP never/RB got/VBD around/\_ to/TO joining/VBG.
- All/DT we/PRP gotta/VBN do/VB is/VBZ go/VB around/\_ the/DT corner/NN.
- Chateau/NNP Petrus/NNP costs/VBZ around/\_ 2500/CD.

Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture 4      11 / 18

Say any new approach should be able to decide what part of speech tag is correct. So, yeah one thing is that it might even be difficult for people in some cases right it is not a very very easy problem always. So, here in this slide we have 3 sentences this is Shaefer never got around to joining all we go to do is to go around the corners and chateau Petrus costs around 2500 and all the 3 words, 3 sentences, you have the word around and you will find out what is the part of speech tag and you will see this is not very very easy if you will just look at these sentences it is not easy to find it what are speech tags.

I will just suggest that you go to the tutorial once and so the tutorial that I talked about earlier. So, that talks about what are the differences between various part of speech that will give you some idea on how to find out the actual part of speech tags of around

in this case. So, what we will see here? Before in the first case, it is a, the particle second case it is a preparation and third case, it is in adverb.

(Refer Slide Time: 16:30)

*Relevant knowledge for POS tagging*

*The word itself*

- Some words may only be nouns, e.g. *arrow*
- Some words are ambiguous, e.g. *like, flies*
- Probabilities may help, if one tag is more likely than another

*Local context*

- Two determiners rarely follow each other
- Two base form verbs rarely follow each other
- Determiner is almost always followed by adjective or noun

Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture 4      13 / 18

Now, what is the relevant knowledge that we might need for this part of speech tagging problem that we might need to give to our models? So, for example, some words might always be in one part of category like arrow is always a noun. So, I can have this knowledge some words are ambiguous like fly, flies and what like it can it is again ambiguous. So, what might help is what is the probability that a word occurs as a particular part of speech tag the baseline that you are talking about this might be helpful for our model also that given a word what is the most probable tag for this word this is one thing that we might use.

Now, what can be the other information that is useful? So, take the word leg flies, if you do not give me any other word, I may never be able to tell with full confidence what should be the corresponding correct part of speech tag unless you tell me the sentence where it occurs. Same was with the word like saw that was one of the earlier examples, unless you give me the sentence like Peter saw her I cannot tell the saw is actually a verb and not a noun so; that means, I need I need to do something of the word itself how come how common it occurs with some part of speech tag then another, I also need to know about the context in which the word is occurring and how can I use the context to segregate the actual part of speech tag. I need the local context in the sentence.

So, for example, the information that determiners really follow each other so if my model is saying 2 words, the determiner which is not allowed. Similar 2 based forms of the word they do not follow each other they should not come together similarly my model can tell me that a determiner is always followed by an adjective or now this can be useful information that if the previous word is a signed determiner it is a highly likely the next row will be adjectives or noun. So, all this we want to encode using our models as well.

(Refer Slide Time: 18:45)

*POS tagging: Two approaches*

**Rule-based Approach**

- Assign each word in the input a list of potential POS tags
- Then winnow down this list to a single tag using hand-written rules

**Statistical tagging**

- Get a training corpus of tagged text, learn the transformation rules from the most frequent tags (TBL tagger)
- Probabilistic: Find the most likely sequence of tags  $T$  for a sequence of words  $W$

Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture 4      14 / 18

What are the various approaches? So we will try to use all this knowledge, but what are the various approaches that we can use? So, for all the problems that we will be dealing with, in this course mostly, you can always use a rule based approach. So this visual some of the earlier models that work that were used in NLP. So, we are given a problem some language will sit together find out what are the symbol patterns or simple kind of if then else rules that one can write down to solve this particular problem.

What would be a rule based solution to this problem for doing the part of speech tagging disambiguation problem. So, what I will do? I find out for all the words that can have multiple part of speech tags what is one particular thing that that can help me decide whether to use one tag or the other and this and given a new context I can again use this is that particular contexts appearing here or not; then I can have a statistical tagging that is I get a training corpus. So, this is a standard model. So, I have a corpus training corpus

that has the tag text by tag text I mean I have the sentence and with each word I also have the actual part of speech category.

Now, using that can I learn, what is the actual part of speech tag for each individual word and this in a new sentence, can I learn some model? So, this is my statistical tagging. So, they are again different various models. So, one simple model is TBL tagger that was one of the earlier model proposed for part of speech tagging. So, that is I am given a any corpus of tag text can I learn some sort of transformation rules that this word has a particular main category most probable category of part of speech tag, but given the context if this most probable text should be changed to some other tag. So, can I use some rules from a corpus training corpus itself? And then the probabilistic models where I will have a probabilistic in the position of what is the most likely sequence of tags for a sequence of words.

(Refer Slide Time: 21:33)

TBL Tagger

*Label the training set with most frequent tags*

- The can was rusted.
- The/DT can/MD was/VBD rusted/VBD.

*Add transformation rules to reduce training mistakes*

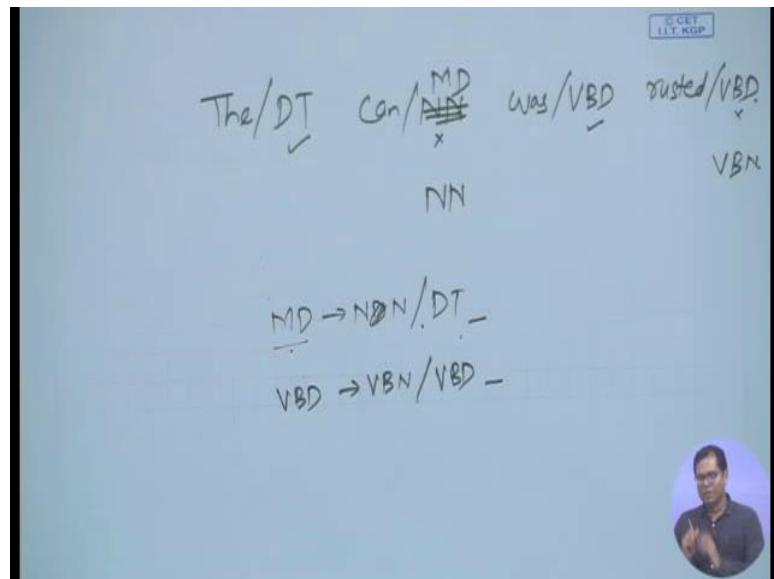
- MD →NN: DT\_-
- VBD→VBN: VBD\_-

We will go through TBL tagger very very briefly and then we will devote a lot of time on the probabilistic taggers. So, what is TBL tagger transformation based learning? So, let us try to understand the idea first. So, I have a sentence like - the can was rusted. So, how the model will process? So, this is my training data set. So, I also know, what is the actual part of speech tag for each of the individual word?

Now, the first thing you will do is to find out for each individual word, what is the most likely part of speech tag. So, in other words the can was rusted. So, if you find out a

more slightly tag for the word can it will be a model work and mostly occurs as a model work similarly last word rusted it would be a verb in the past tense that is the most popular probable tag.

(Refer Slide Time: 22:24)



I will write down the can was rusted, now I will see in my actual training sentence what are the actual or thus rejected are assigned to different words. So, I find this is correct this is incorrect this I am sorry, this the most probable tag will be MD moreover is incorrect in this and as I will have a noun this is fine, but this will be instead VBN is a participant past participle. So, these 2 are incorrect. So, I need to write some moves so, that these can be transformed.

One symbol can be MD changes to NN sorry to NN when preceded by DT remember the rules that we saw in the previous lecture. So, it needs to be if preceded by something and there is nothing so, we are not putting any restriction what is being followed. So, this is a rule that I can use. So, now, what will happen? In a new case whenever a word is assigned MD, if I see the previous word is determiner I will change MD to NN, this is the rule I am running from this example similarly I can rule a learn, a rule here VBD goes to VBN. Whenever preceded by reading this can be another rule.

So, this is the idea - I have training corpus I find out for each for what is the most likely tag whenever they do not match I will write down some set of rules and then I keep on

doing that and I might put down a separate data set, simple small data set for testing how good my finally, tagger text. So, this is my TBL tagger. So, this is what we have seen.

(Refer Slide Time: 24:21)

**Probabilistic Tagging: Two different families of models**

**Problem at hand**  
We have some data  $\{(d, c)\}$  of paired observations  $d$  and hidden classes  $c$ .

**Different instances of  $d$  and  $c$**

- **Part-of-Speech Tagging:** words are observed and tags are hidden.
- **Text Classification:** sentences/documents are observed and the category is hidden.  
Categories can be positive/negative for sentiments ..  
sports/politics/business for documents ...

**What gives rise to the two families?**  
Whether they generate the observed data from hidden stuff or the hidden structure given the data?

Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture

Now, what is probabilistic tagger? In probabilistic tagging I will have a probabilistic interpretation of what is the most likely sequence of tags that should be used for a given sentence. So, let me just briefly talk about what in general the probabilistic tagging models do. So, in these models we have some data that is some observations  $d$  and a certain class. So, in the case of part of speech tag what is the example of  $d$  and  $c$ ? So,  $d$  are the words and then tags are my classes I want to assign various classes that that tags to different words. So, tags some classes.

Now, you take a different problem like text classification what will happen the documents all the sentences if you are doing classification or documents a sentence is they become your data and you have your classes. So, suppose using sentiment analysis. So, the sentence is your data and the class is positive negative or neutral if you are doing text classification in terms of sports, visage, politics and so on, there is categories, the document or the vertical instance of the text becomes your data and that becomes your class a sports and politics and so on. So, you have a paired observation of a data and a class.

Now, they are 2 different families of probability models that can be used for solving these problems I will just briefly talk about these 2 families and we will take examples

from both of these for this problem. So, and you will be able to use that for many other applications NLP also. So, what gives rise to 2 different families that is whether you generate the data from the class or the class from the data. So, what is the philosophy of your model?

(Refer Slide Time: 26:14)

*Generative vs. Conditional Models*

**Generative (Joint) Models**

Generate the observed data from hidden stuff, i.e. put a probability over the observations given the class:  $P(d,c)$  in terms of  $P(d|c)$   
e.g. Naïve Bayes' classifiers, Hidden Markov Models etc.

**Discriminative (Conditional) Models**

Take the data as given, and put a probability over hidden structure given the data:  $P(c|d)$   
e.g. Logistic regression, maximum entropy models, conditional random fields

SVMs, perceptron, etc. are discriminative classifiers but not directly probabilistic

Pawan Goyal (IIT Kharagpur)      Introduction to POS Tagging      Week 3: Lecture

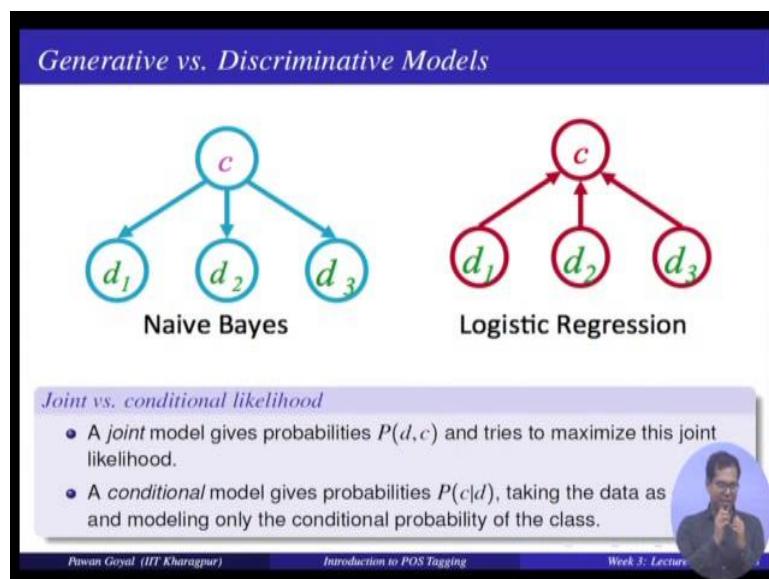
Let us try to understand that from this slide, so, we have 2 different types of models, one is a generative model and second is a conditional model. So, in generative model what will happen? So, you have appeared data and the class. So, generative model you will assume that the class is there and the class generate the data. So, this is the philosophy from the class that it I generated. So, examples are like nigh page modular hidden Markov models. So, your class is given and data generated from this.

In the discriminative condition models what will happen, that your data is there and you assume that hidden state is a generated from the data. So, that is the minor difference between that. So, in the condition, in the joint model you first at the class you first generate the class and then you generate the data from the class. So, that is so to give a simple example if you want to use a joint for generating modules for text classification what will you assume, if I am going to take a document I will first think over what is the topic I want to write say politics I think about the class and now given this class what is the probability that I write all these words. So, from that class I find out the probability of different different words. So, that is how my model is defined. In the case of

conditional model given the observation directly I want to find out the probability of the class.

This difference tells me whether so how do I actually go about solving these models. So, they are the models like SVMs perceptron that are not probabilistic. So, they are not in division one of these. So, they are also discriminative classifiers.

(Refer Slide Time: 28:05)



Now, this picture will help you understand that 2 models again. So, you see the directions are different in the pictures in the first direction, in the first in the left hand side picture. So, direction is from the class to all the data points. So, first the class is generated then all the data points so you generative model.

In the second one is a conditional model. So, given the data you want to find out directly the probability of the class. So, first one example is Naive Bayes, second one example is logistic regression. So, a joint model will give you probability of,  $d_1$  the data and the class together and you will try to maximize the joint probability and condition of model you will directly want to find out the probability of the class given a rate. So, we will take examples of both of these in the next modules. So, what is a joint model that can help me solve the part of speech tagging problem and what is the condition model that can help me solve the part of speech tagging problem.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute Technology, Kharagpur**

**Lecture - 16**  
**Hidden Markov Models for POS Tagging**

Hello everyone, welcome back friend for this lecture 5 of week 3. So in the last lecture we started with the problem on part of speech tagging. We defined the problem image given a text that can be a sentence. So you need to find out what is the actual part of speech category for each of the individual word. So there may be many various ambiguities, but you need to resolve the ambiguities and find out the unique part of each tag for each of the words.

And we said that you can solve it using rule based methods or some probabilistic methods. We also discuss that the methods can be generative or discriminative. And then they differ in the philosophy of the model. So in general generative model the class comes first and it is assumed that the words are generated the data is generated from the class. And in the discriminative model you directly find out the probability of the class given the data. So in this lecture we will start with the generative model that is hidden Markov model and see how it can be used for solving the task of part of speech tag. So this is a probabilistic model.

(Refer Slide Time: 01:40)

*Probabilistic Tagging*

- $W = w_1 \dots w_n$  - words in the corpus (observed)
- $T = t_1 \dots t_n$  - the corresponding tags (unknown)

*Tagging: Probabilistic View (Generative Model)*

Find

$$\begin{aligned}\hat{T} &= \operatorname{argmax}_T P(T|W) \\ &= \operatorname{argmax}_T \frac{P(W|T)P(T)}{P(W)} \\ &= \operatorname{argmax}_T P(W|T)P(T) \\ &= \operatorname{argmax}_T \prod_i P(w_i|w_1 \dots w_{i-1}, t_1 \dots t_i)P(t_i|t_1 \dots t_{i-1})\end{aligned}$$

Pawan Goyal (IIT Kharagpur)      Hidden Markov Models for POS Tagging      Week 3: Lecture 5      2 / 17

$$\hat{T} = \operatorname{argmax}_T P(T|W)$$

$$= \operatorname{argmax}_T P(W|T)P(T)/P(W)$$

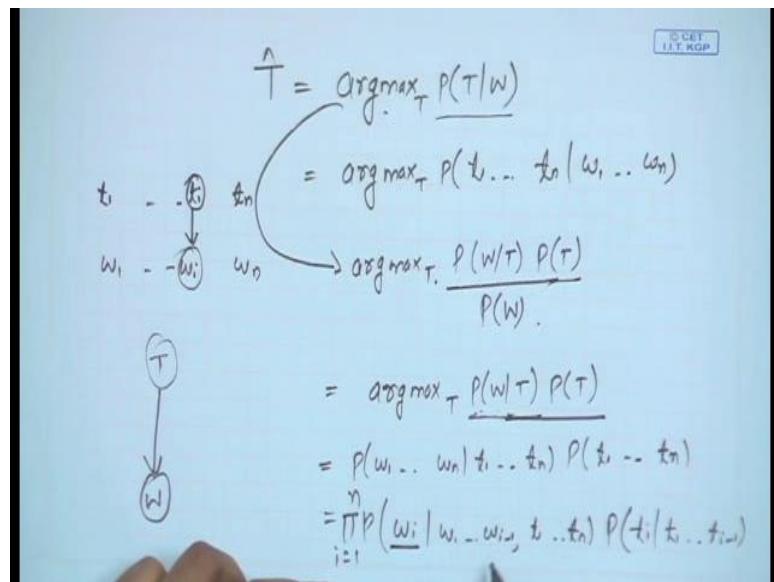
$$= \operatorname{argmax}_T P(W|T)P(T)$$

$$= \operatorname{argmax}_T \prod_i P(w_i|w_1, \dots, w_{i-1}, t_1, \dots, t_i)P(t_i|t_1, \dots, t_{i-1})$$

So starting with what is my problem. So I have some  $n$  words  $w_1$  to  $w_n$  in my corpus that I observed in, and I need to find out the participation text for each of these words. So suppose that you have to find out the sequence capital  $T$ , to that assigns  $t_1$  to  $t_n$  for all of these  $n$  words. Now each of these  $t_i$ 's that a participant text can belong to my actual say suppose I am using the university of when we tag set. So it can take any of those 45 value age. So there are many different values that these this sequence can take. I need to find out the actual and ambiguous tag sequence given this word sequence as my input.

So now how do I start solving this problem as per this probabilistic model? So the idea is that among all the possible sequences of part of speech tags that this word sequence can take I need to find the one that has the maximum probability. So I can write it like that.

(Refer Slide Time: 02:51)



So this is I have to find out that that gives you the maximum probability argmax overall possibility probability  $T$  given  $W$ . So I need to find out the purpose sequence that gives the highest probability.

Now, so as we have already said  $T$  is nothing, but  $t_1$  to  $t_n$  and  $W$  is nothing, but  $w_1$  to  $w_n$ . So I can write it as argmax  $t_1$  to  $t_n$  given  $w_1$  to  $w_n$  this is what I will be doing. So in generative model remember what is the idea, idea is that the class or the tags come first and then my words are generated from there. So I cannot find out the probability of tagging the word, but from the model I can find out the probability of word given the tag. So I need to invert this, the direction of the probabilities here. So instead of finding  $t$  given  $w$  i need to find I need to use  $w$  going.

So what is the theorem popular theorem that I can use I can apply Bayes theorem. So instead of directly opening it here I can try same argmax  $T P W$  given  $T P T$  given divided by  $P W$  and  $P W$  it is common for all these sequences the probability of the sequence. So this is again it does not matter. So that will give me argmax over  $T P W$  given  $T$  and  $P T$ . So this because a using generating model. So first my class that is my  $T$  comes and then my sequence is generated  $W$  that is why I have to take it in this format probability  $W T P T$ .

Now I can try and open this. So this will be. So let me just take this particular thing and this will be the probability  $w_1$  to  $w_n$  given  $t_1$  to  $t_n$  probability  $t_1$  to  $t_n$ . So now, so I can again use chain rule to write that, so that we can see in the slide. So I can write it as using the chain rule. So this will be nothing, but probability. So multiplication over all  $i$  is equal to 1 to  $n$ ,  $w_i$  given  $w_1$  to  $w_{i-1}$  minus 1  $t_1$  to  $t_n$  and probability  $t_i$  given  $t_1$  to  $t_{i-1}$  minus 1 this is simply by using the chain rule I can write it like that.

So now, it is very difficult to get the estimates for all these probabilities that we are thinking it. That means, I need to do certain simplifications over this formula. So what are the simplifications that we do? So one simplification that we do here is that, so in this formula we are saying that the probability of the word the current word. So you see we have a sequence of words  $w_1$  to  $w_n$  and correspondingly we have a sequence of  $x t_1$  to  $t_n$  whatever  $w_i$  what I am saying the probability of this word depends on all the previous words and all the tags. So instead of that because the generative model I might

say that this probably depends only on the current deck, this is simplification that I can make.

(Refer Slide Time: 07:18)

$$\begin{aligned}
 & P(w_i | t_i) = \arg \max_{w_i} P(w_i | t_i) \\
 & \Rightarrow \arg \max_{t_i} \frac{P(w_i | t_i) P(t_i)}{P(w_i)} \\
 & = \arg \max_{t_i} P(w_i | t_i) P(t_i) \\
 & = P(w_1, \dots, w_n | t_1, \dots, t_n) P(t_1, \dots, t_n) \\
 & = \prod_{i=1}^n P(w_i | t_i, \dots, t_{i-1}) P(t_i | t_{i-1}, \dots, t_1) \\
 & \approx P(w_i | t_i) \quad P(t_i | t_{i-1}) \\
 & \quad \text{[bigram assumption]}
 \end{aligned}$$

So this I can simplify as probability  $w_i$  given  $t_i$ . So this is one simplification, then here we are saying that the probability of  $t_i$  the tag  $t_i$  depends on all the previous tags. So again I might simplify it by using some Markov assumption that it depends on either the only the previous tag or previous to previous tag. So if I take only the bigram assumption. So I will set  $t_i$  depends on  $t_{i-1}$ . So in terms of this model I will say that  $t_i$  only depends on  $t_{i-1}$  and here I will simplify it using probability  $t_i | t_{i-1}$  and this is my bigram assumption. So now if I make the simplification what is the model that we actually see?

(Refer Slide Time: 08:20)

*Further simplifications*

$$\hat{T} = \operatorname{argmax}_T \prod_i P(w_i|w_1 \dots w_{i-1}, t_1 \dots t_i) P(t_i|t_1 \dots t_{i-1})$$

- The probability of a word appearing depends only on its own POS tag  
 $P(w_i|w_1 \dots w_{i-1}, t_1 \dots t_i) \approx P(w_i|t_i)$
- Bigram assumption: the probability of a tag appearing depends only on the previous tag  
 $P(t_i|t_1 \dots t_{i-1}) \approx P(t_i|t_{i-1})$
- Using these simplifications:  
 $\hat{T} = \operatorname{argmax}_T \prod_i P(w_i|t_i) P(t_i|t_{i-1})$

Pawan Goyal (IIT Kharagpur)      Hidden Markov Models for POS Tagging      Week 3: Lecture 1

$$\hat{T} = \operatorname{argmax}_T \prod_i P(w_i|w_1 \dots w_{i-1}, t_1 \dots t_i) P(t_i|t_1 \dots t_{i-1})$$

So this is the formula that we came up with. So I want a tag sequence that has the highest probability for this particular form and we make some simplification that is the probability of a word appearing depends only on it is part of speech tag remember this is genetic model. So the word is generated from the part of speech tag. So first the part of speech tags is generated and then the tags are giving each an individual word. So we are making this assumption that the word is generated only from its own part of speech tag then. So this gives me the first simplification second one I will say that the probability of attack depends only on its forward previous tags.

So if I make the bigram assumption, it will depend only on the previous tag. So this will give me this function. So together the same the simplification will give me this formula. So I want to find it tag sequence that gives me the maximum probability for this particular formula. So now, once we have come to this formula. So what is this model actually?

(Refer Slide Time: 09:29)

*Computing the probability values*

*Tag Transition probabilities  $p(t_i|t_{i-1})$*

$$P(t_i|t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$
$$P(NN|DT) = \frac{C(DT, NN)}{C(DT)} = \frac{56,509}{116,454} = 0.49$$

*Word Likelihood probabilities  $p(w_i|t_i)$*

$$P(w_i|t_i) = \frac{C(t_i, w_i)}{C(t_i)}$$
$$P(is|VBZ) = \frac{C(VBZ, is)}{C(VBZ)} = \frac{10,073}{21,627} = 0.47$$



Pawan Goyal (IIT Kharagpur)      Hidden Markov Models for POS Tagging      Week 3: Lecture 1

So first let us see that it can be easily compute these probabilities. Probability  $w_i$  given  $t_i$  and probability  $t_i$  given  $t_{i-1}$ , so how will you actually compute these probabilities? So one way is that you are given a corpus where you know all the words you also know what are their part of speech tags somebody has manually annotated each this, this data file, now given this data can you compute these probabilities. So for example, computing probability  $t_i$  given  $t_{i-1}$ . If you want to find out how many times this tag  $t_i$  comes after the tag  $t_{i-1}$ . So you will compute it by using the maximum likelihood estimate that is the number of times  $t_{i-1}$  and  $t_i$  come together in the corpus divided by the number of times the word  $t_{i-1}$  the tag  $t_i$  minus 1 comes. So if you see here in the slide. So  $p(t_i|t_{i-1})$  can be found by this by using these counts, count of the 2 texts together divided by the count of the previous tag only.

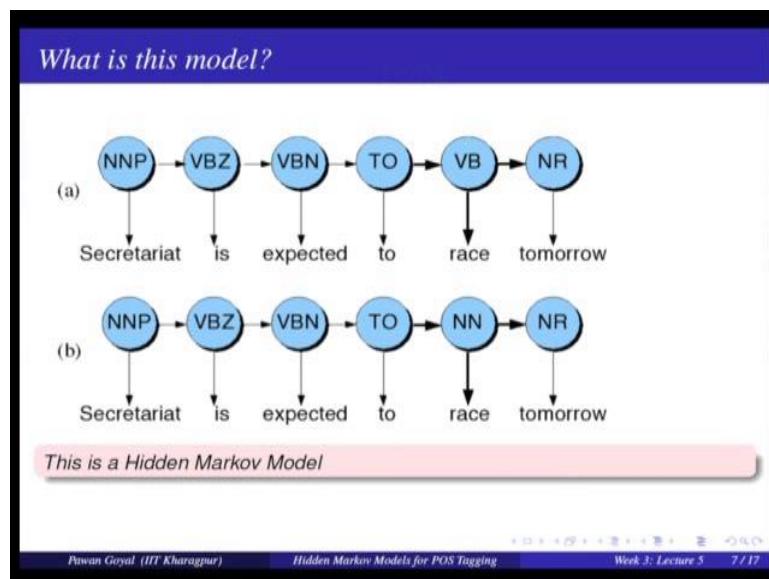
$$P(t_i|t_{i-1}) = C(t_{i-1}, t_i)/C(t_{i-1})$$

So if I want to compute probability  $N, N$  given  $DT$  I will say number of times  $DT$  occurs followed by  $CN$  and  $N$  divided by number of times  $DT$  occurs. And this if I have the numbers I can compute this probability. What is the other probability of the compute? I have to compute probability of word given the tag. So the complete probability where I give the tag I will find out again how many times this word occurs with this tag divided by how many times that tag occurs, so this again from my corpus.

So if I have the numbers I want to compute probability is given V w Z I will find out how many times the word each occurs with the direct V w Z divided by the number of times the tag V w Z actually occurs in my corpus. So here if I have the numbers I can compute these probabilities. So all the probabilities that are required for this model can easily be computed if I have a data where I know the words and the part of speech tags for each and individual words. )

$$P(w_i | t_{i-1}) = C(t_i, w_i) / C(t_i)$$

(Refer Slide Time: 11:39)



So now let us see how we can use that for some disambiguation. It is not the complete model just to give you some idea. So once we have this how we can use that for some disambiguation. So I have a sentence a part of a sentence here. Secretariat is expected to race tomorrow. And the ambiguity here is in the word race. So whether the other word races in noun NN or VB you see everything as the same here. Now what are the probabilities as from my model that differ in the 2 interpretations.

(Refer Slide Time: 12:15)

Inter. 1	Inter 2
$P(VB To) *$	$P(NN To) *$
$P(NR VB) *$	$P(NR NN) *$
$P(race VB)$	$P(race NN)$
= ✓	= ✓

$P(w|i)$        $P(t|b)$   
Chirag

So if you see the first interpretation, what is the second interpretation? What are the probabilities that are defined, in the first one you find if the probability of the tag VB given 'To' versus in the second you find probability of NN given 'To' then in the first when you find probability of NR given VB and second you find probability of NR given NN. In the first when you find probability of race given VB and second you find probability of race given NN and because in your model you are multiplying all the probabilities; that means, you will multiply all the prob all these 3 probabilities in the interpretation 1 and all 3 in the interpretation 2.

So whichever multiplication gives you the highest value will decide what is the actual part of speech tag of race that should be used here. It should be VB or NN. So if you have the corpus you know all the probabilities you will multiply you will find the number and this will tell me whether I should prefer interpretation 1 or interpretation 2.

(Refer Slide Time: 13:33)

The slide has a blue header bar with the title "Disambiguating 'race'". Below it is a purple box containing the text "Difference in probability due to" followed by three bullet points:

- $P(VB|TO)$  vs.  $P(NN|TO)$
- $P(race|VB)$  vs.  $P(race|NN)$
- $P(NR|VB)$  vs.  $P(NR|NN)$

Below this is a green box containing the text "After computing the probabilities" followed by two bullet points:

- $P(NN|TO)P(NR|NN)P(race|NN) = 0.0047 \times 0.0012 \times 0.00057 = 0.0000000032$
- $P(VB|TO)P(NR|VB)P(race|VB) = 0.83 \times 0.0027 \times 0.00012 = 0.00000027$

At the bottom of the slide, there is a footer bar with the text "Pawan Goyal (IIT Kharagpur)", "Hidden Markov Models for POS Tagging", "Week 3: Lecture 5", and "6 / 17".

Suppose I take some numbers. So here so difference is because of these probabilities suppose from my corpus I find some numbers. So I will see here is that the possibility here is that the word race should be a verb not as a noun. And if you see the sentence to race tomorrow the race should be race, the word race should be used as a verb not as a noun, although noun is a more common category count part of speech tag for race then verb.

But because if the context it is more likely to have race edge verb in this case. So that is how we can disambiguate in one part in this simplistic case, but in general for the whole sentence even if you do not know any of the tags we can try to use this model to find out what is the actual sequence of tag that should be used.

So now so coming back to this question what is this model. So what are you seeing here? So you are seen, so you have a sentence. So that is the words that is what you are observing and then there are certain tags that assigned to each of the individual words. So in this model you have a probability of going from one tag to another tag. And then from a tag you get a word. So can you find can you think of what is the model it corresponds to what is the actual model. So if you have come across hidden Markov model where you have the states and from one state you transit to another state and so on and in each state that is hidden you can emit the observation. So the word here is observation. So this is nothing, but the hidden Markov model.

(Refer Slide Time: 15:23)

- Tag Transition probabilities  $p(t_i|t_{i-1})$
- Word Likelihood probabilities (emissions)  $p(w_i|t_i)$
- What we have described with these probabilities is a hidden markov model.
- Let us quickly introduce the Markov Chain, or observable Markov Model.

So what are hidden Markov models? So just to give the idea in brief, here you have the tag transition probabilities,  $t_i$  given  $t_{i-1}$  you have the emission  $p(t_i|t_{i-1})$

probabilities. So that is word observation probabilities probability  $w_i$  given  $t_i$ . So using this, whatever we are describing is a hidden Markov model. Now to tell you what is hidden Markov model? So let me just quickly tell you what is a Markov model and how a (Refer Time: 15:53) Markov model different from a Markov model.

(Refer Slide Time: 15:58)

*Weather example*

- Three types of weather: *sunny*, *rainy*, *foggy*
- $q_n$ : variable denoting the weather on the  $n^{\text{th}}$  day
- We want to find the following conditional probabilities:  
$$P(q_n|q_{n-1}, q_{n-2}, \dots, q_1)$$

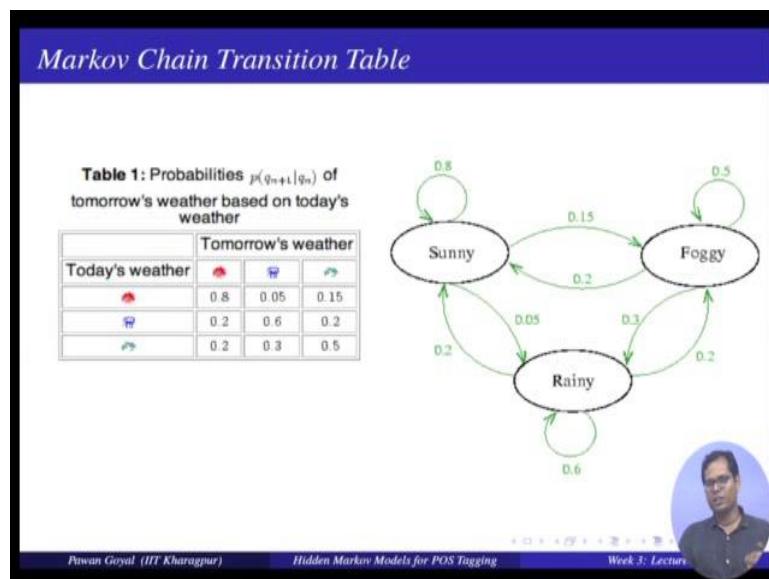
*First-order Markov Assumption*

$$P(q_n|q_{n-1}, q_{n-2}, \dots, q_1) = P(q_n|q_{n-1})$$

So what is a Markov model? So this Markov model is best explained using this simple example. So in Markov model what happens you again have states, but the states are also your observations. Suppose you are studying the weather the weather on the current day and the weather can be sunny, rainy or foggy. These are the 3 different kinds of weathers that can happen on a given day. Now what you will get you will have these 3 states you will know, given that today is sunny what is the probability that tomorrow will be sunny or foggy or rainy. So this is state transition probabilities you will obtain. So suppose  $q_n$  is a variable that denotes the variable on the nth day and using this model we can find out the probability of  $q_n$  the weather on the nth day given all the previous days back and here if you use the first row mark Markov model. So we say this will depend only on the previous days' back  $q_n$  will depend on your  $n - q_n - 1$ .

$$P(q_n|q_{n-1}, q_{n-2}, \dots, q_1) = P(q_n|q_{n-1})$$

(Refer Slide Time: 17:03)



So let us take one simple example. So here you are seeing the state transition. So that says that if today's weather is sunny then tomorrow will be sunny with the probability of 0.8 and if today is sunny tomorrow will be foggy with a probability 0.15. So you can see that from the edges that go from one state to another state, also from the table that is shown in this. So now, once you are given these probabilities you can do certain computations.

(Refer Slide Time: 17:35)

### Using Markov Chain

- Given that today the weather is sunny, what is the probability that tomorrow is sunny and day after is rainy?



Pawan Goyal (IIT Kharagpur)      Hidden Markov Models for POS Tagging      Week 3: Lecture

For example, suppose you have to find out given that today the weather is sunny what is the probability that tomorrow is sunny and day after is rainy.

(Refer Slide Time: 17:48)

$$\begin{aligned} & P(q_{n+2} = \text{rainy}, q_{n+1} = \text{sunny} \mid q_n = \text{sunny}) \\ &= \underbrace{P(q_{n+1} = \text{sunny} \mid q_n = \text{sunny})}_{0.8} \cdot \underbrace{P(q_{n+2} = \text{rainy} \mid q_{n+1} = \text{sunny}, q_n = \text{sunny})}_{\approx 0.05} \\ &\quad \text{Observation} \\ &\quad \begin{array}{c} \textcircled{1} \\ \textcircled{2} \end{array} \quad \begin{array}{c} q_1 \\ q_2 \\ q_3 \end{array} \quad = 0.04 \end{aligned}$$


So let us use the variable and I want to find out probability  $q_n$  plus 2, days after is rainy  $q_n$  plus 2 is to rainy and tomorrow is sunny  $q_n$  plus 1 it is sunny given  $q_n$  sunny.

So how would you compute that? So if you simply are the channel, you will say that is nothing, but probability  $q_n$  plus 1 is equal to sunny given  $q_n$  is equal to sunny times

probability  $q_n + 2$  is equal to rainy given  $q_n$  is equal to sunny  $q_n + 1$  is equal to sunny and now because you are using a first row Markov assumption. So this will be equivalent to. So this you will compute using probability  $q_n + 2$  is equal to rainy given  $q_n + 1$  is equal to sunny. So, now you have to compute this probability and this probability and that you can obtain from the state transition graph. So if you go to the previous slide you will find out probability sunny given sunny is 0.8 m probability rainy given sunny is 0.05. So once you multiply this you will get the answer is 0.04. So here so you will get the answer is 0.04. So that gives you the probability that tomorrow will be rainy and sorry; tomorrow will be sunny day after tomorrow will be rainy given that today is sunny. So that is how you use the Markov model.

Now, this is the Markov model. So what you see here? You have the states, so in this case the weather. So on a day is a solidus state and transitions are happening among the states, but what is your observation that is also state. So you also observing the weather that is shear state you are observing that now remember the example of participate text. What are we observing? We observing the words, then when I give you a text I you only observe the words, but what is hidden that is the text. The text is hidden. So that is where the hidden Markov models are different from Markov model.

The states there are not observed, they are hidden variables and what is observed is different. So you have the words are being objective and from the state you can emit the words. So this is the idea. So the generative model would be. So you are starting from some state 1 you keep on transiting to other states S3 and so on and v is a state you also emit a word emission 1, emission 2 and so on and these emissions are nothing, but your observations, these are your observations. So you need to find out the underline sequence of a states given the sequence of observations.

(Refer Slide Time: 20:54)

**Hidden Markov Model**

- For Markov chains, the output symbols are the same as the states  
*'sunny' weather is both observable and state*
- But in POS tagging  
*The output symbols are words*  
*But the hidden states are POS tags*
- A Hidden Markov Model is an extension of a Markov chain in which the output symbols are not the same as the states
- We don't know which state we are in

Pawan Goyal (IIT Kharagpur)      Hidden Markov Models for POS Tagging      Week 3: Lecture 5      12 / 17

So now this is the difference we have seen for Markov mode chains the output symbols are the same as the states. So the word sunny is both a state and the observable. So what happens in part of speech tagging, words are the output symbols, but the hidden states are part of speech tag. So hidden Markov model is nothing, but an extension of Markov of chain, so in which what happens that the output symbols that you are having are not the same as the states. So states are different from the output. And we actually do not know what should we are in until we try to use our model.

(Refer Slide Time: 21:33)

**Hidden Markov Models (HMMs)**

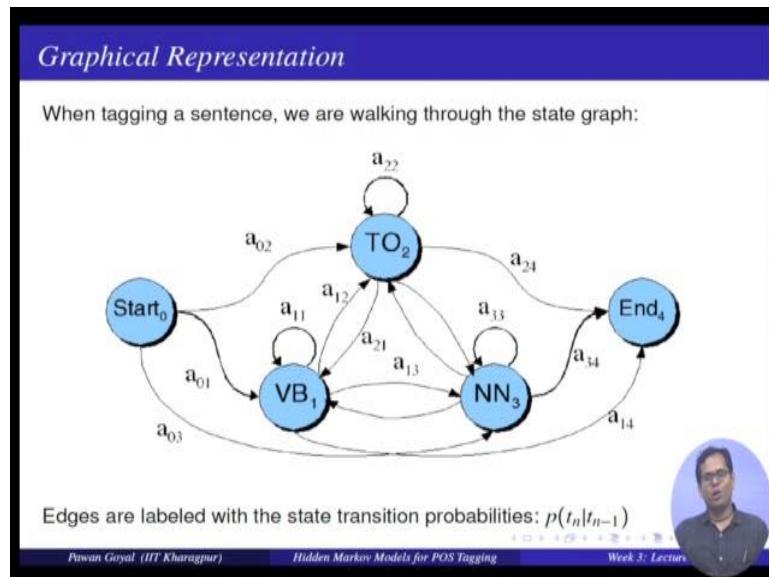
*Elements of an HMM model*

- A set of states (here: the tags)
- An output alphabet (here: words)
- Initial state (here: beginning of sentence)
- State transition probabilities (here  $p(t_n|t_{n-1})$ )
- Symbol emission probabilities (here  $p(w_i|t_i)$ )

Pawan Goyal (IIT Kharagpur)      Hidden Markov Models for POS Tagging      Week 3: Lecture 5      13 / 17

So what are the elements of a hidden Markov model? So what do you need from hidden Markov model? So you need a set of states, yes. You need the probability of transiting from one state to another state. In the probability of emission given a state which words will be emitted with what probability. And you might also need what is the beginning of a state and so on. So if you try to correspond a hidden Markov model with part of speech is tagging. So we need set of states in our part of speech taking case. So the tags are the states. We need an output alphabet that is then what are emissions. So the words are the emissions with the initial state. So in our case that is the beginning of the sentence. We need the transition probabilities that is given a previous tag what will be the next tag. And we need the emission probabilities that is given this tag what will be the word.

(Refer Slide Time: 22:40)

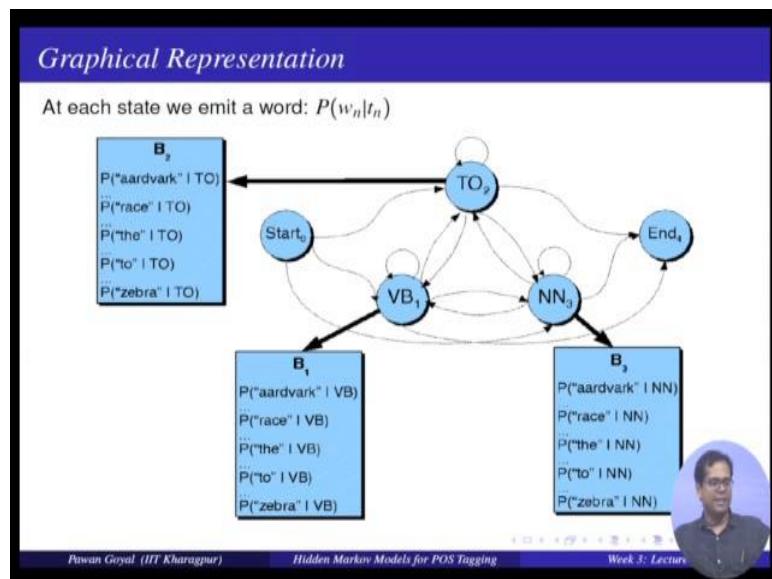


So once we have this we can also give a graphical representation to our hidden Markov model for part of speech tagging. So what is happening? See here we start state. So here by start steady shown. From the start state you have a probability of transiting to any of the other tags. So what does that mean? You can start the sentence with some particular tags. So this probably should depend on what is the tag that is more likely to start the sentence. And once you have this tag what is the next likely tag and so on. So this is your state transition graph. So this is what is shown in the slide. So when we are tagging the sentence you are actually walking on this a state graph from one state we are going to another state and so on and they are very transition probabilities that you can have over

this state graph. And this you can model by using probability of a state  $t_n$  given  $t_{n-1}$ .

Now, what else is missing here? With each now within the hidden Markov model with each a state you also have the word emission. So from history you also want to know what is the probability that you will output or emit a particular word.

(Refer Slide Time: 23:52)

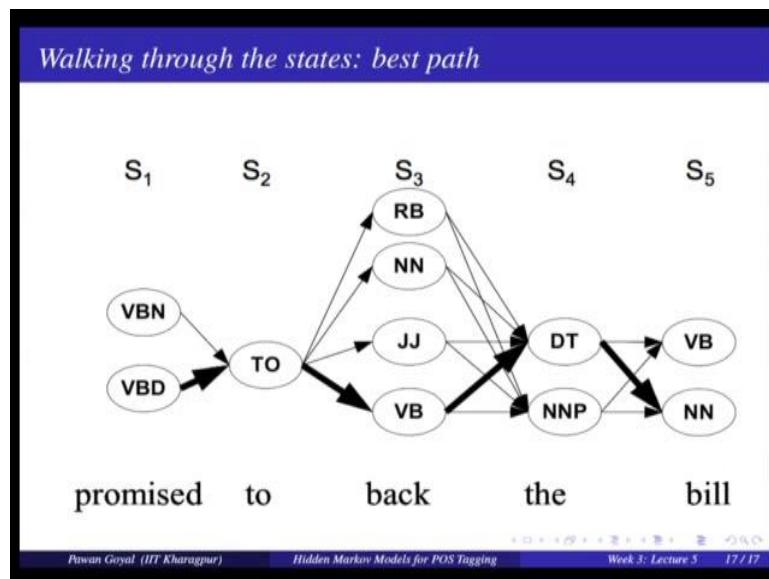


So this is what is additionally needed here. So in the graphical representation, now with these states like  $TO$  I should also the probability of different words emitting from that state. So here all the words are in the vocabulary are written and given the particular state what is the probability of out putting or emitting that particular word in the vocabulary, so that you need to define for all the very possible states in your graph.

So now once we have both these what is my problem. So I can define, so if you remember I can define all these probabilities once you give me a corpus that contains a set of words and their corresponding part of speech tags. If you give me that I can find out all these probabilities. So I can define these graph once here once I have a corpus.

What is my problem? At runtime I am only given an observation that is, I am only given a sentence that is sequence of words. I have to find out what is the corresponding part of speech tag sequence that should be used for this word sequence.

(Refer Slide Time: 25:09)



So that is suppose I am given a sentence like this, is the part of the sentence promised to back the bill, there are 5 words here. So my problem is to find out what is the sequence of  $x$  that would be used for this sentence. So how would I approach this problem given the state transition graph that I already have? Now for each word one thing to simplify this I will do is that, for each word I will find out what are all the possible part of speech tags a word can take.

So we will be talked about this said that some words can take multiple part of speech tags, but none of the words went beyond 7 part of speech tags. And mostly the words were having if they were ambiguous they were having 2 or 3 part of space tags. So for a word if I can identify what are the part of speech tags. So I can only use that to have my set of possibilities. So here for example, I will say the word promised can take only VBD or VBN past tense of participial. It can be only one of those. I have to disambiguate among the 2 - the word to can have only one part of speech tag. Back can have 4 tags, the can have 2 tags, and bill can have 2 tags. These are all the possibilities. Now each word can take any of these possibilities. So now, can you just quickly see how many possibilities are there? So if you see here I have 2 times 4, 8 times 2, 16 times 2, 32 possibilities of the part of speech tag sequences.

Now, I have to find out among these which is the most likely sequence of participation tags that is being used. So how will I solve this problem? One neither might be, I

enumerate all the possibilities and for each possibility I compute the probability separately. So I have 32 possibilities and I compute 32 different. So for all the possibility I compute the full probability. That is one possibility, but firstly, this is not very efficient, and think of some sentences which might have say 15-20 words. This will just go exponentially with the number of words. So this is not a good solution.

So I need to have a solution where it does not grow exponentially with the size of the sentence. So what will be a good algorithm for doing that? So ideally I want to come up with the actual part of speech tag sequence like here it will be VBD TO VB DT and NN from all the possibilities. And how will I do that and that is what we will be discussing in the next lecture. So this is the Viterbi algorithm.

How do I apply in my HMM model this will be algorithm to come up with this part of speech tags sequence in efficient manner? Instead of doing something naive implementation that is exponential and that is actually not feasible for doing it for a large corpus, so that will be the focus in the next lecture.

Thank you.

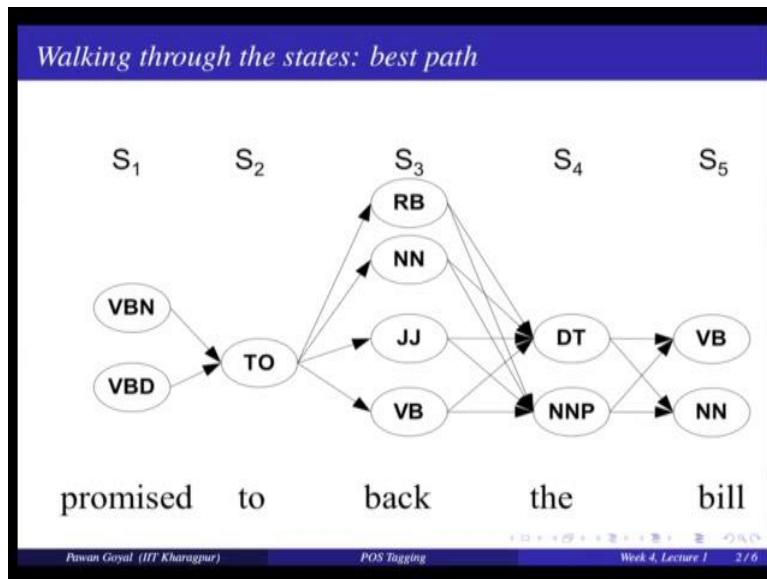
**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 17**  
**Viterbi Decoding for HMM, Parameter Learning**

Welcome back for the 4th week of the course. So, in the last week we had discussed a lot about part of speech tagging and what are the various algorithms as such one can use to solve this problem and we started with one particular classifier that is HMM model for doing part of speech tagging. So, we had gone through the basics of HMM and what are the different probabilities then one need to use to be able to come up with the best sequence given a new sentence. So, what do I mean by sequence given new sentence I want to find out what are the part of speech tags for each of the word in the sentence. So, there are multiple; there are many many possibilities and you need to find out what is the best probable tag sequence. So, we formulated this problem as noisy in the noise channel model framework and then we came up with the hidden Markov model by using certain assumptions.

So, in the last class so we had also discussed a bit about what we will be doing at run time when we are given a sentence, so let me start from there and we will see a particular algorithm on the Viterbi decoding to find out the actual tag sequences in a very very efficient manner and then later on I will go to the parameter learning part. So, we had talked about in the last week also, but we will devote sometime over that in this week.

(Refer Slide Time: 01:53)



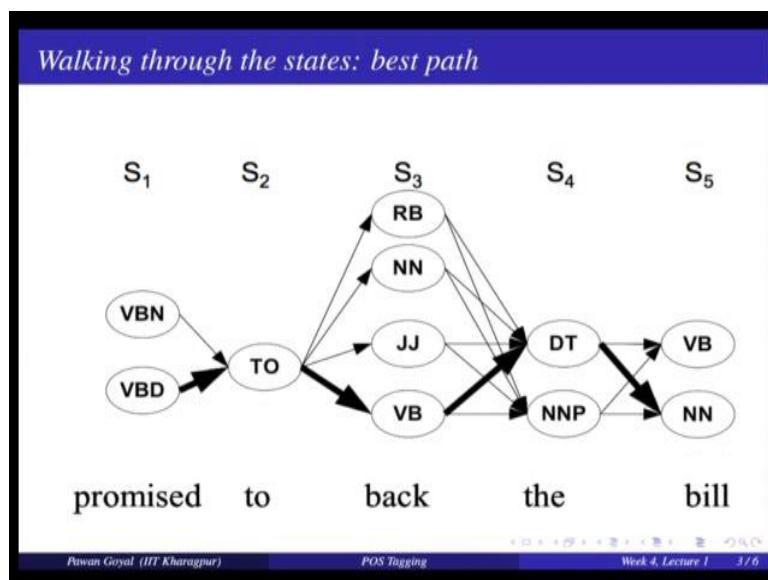
We had seen this figure in the last week itself. So, we have the sentence promised to back the bill and I want to obtain the part of speech tag sequence for the sentence. So, how do I start? I first start by finding out what are the probable part of speech tags for each of the individual words in the sentence. So, further word here promised the 2 possible tags are VBD and VBN, it can be past tense or the participle for the word to there is only one part of speech tag possible for the word back there are 4 part of speech tags possible, we have seen that in one of the lectures for the word the; there are 2 part of speech tags possible for the word bill there are 2 word of speech tags possible. So, I start by enumerating all the possibilities for each of the individual words in the sentence.

Now, I want to find out what is the actual part of speech tag sequence for this word for this sentence? Now looking at this picture, how many different possibilities can you see? So, if you start from any of the part of speech tag for the word promised and go to any of the participant tag for the word bill, any particular path denotes a possible tag sequence. So, if you try to count the number of possibilities so it will be 2 times, 1 times, 4 times, 2 times 2, it will be roughly 32 possible part of speech tag sequences and among the study to; I have to find out, what is the most probable tag sequence.

Now, how should I do that? So, 1 knife method would be that I enumerate all the possible sequences. So, all the 32 sequences I compute the probability for each sequence. So, now, you know how to compute the probability by using the HMM kind of model. We did one

simple example in the last class and find out which of the tag sequence has the highest probability that will be a naive method, you might even do that for if you have say only 32 or 50 possible text sequences, but think about the sentences that might have 10 to 15 words and some words might have 5 to 6 different tags. So, you will see that it will grow exponentially. So, enumerating all the sequences and computing the probabilities is not an efficient solution. So, you might want to do something more efficient.

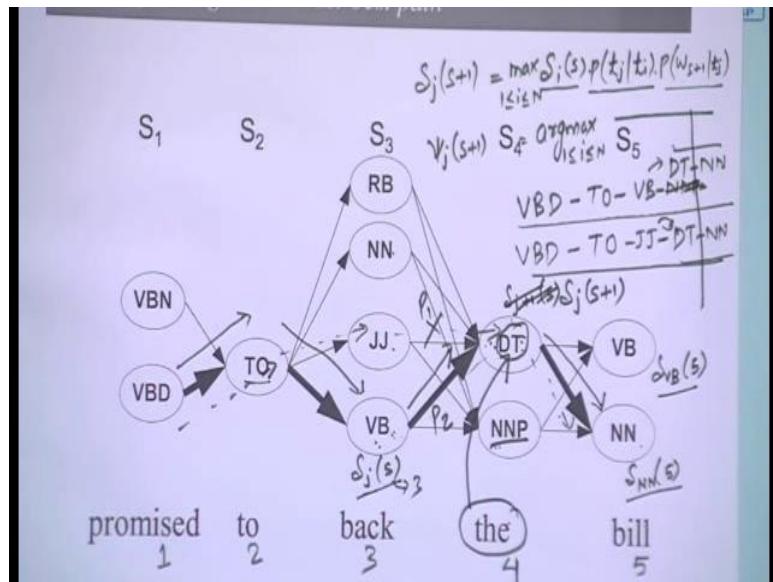
(Refer Slide Time: 04:27)



So that you can find out what is the most probable tag sequence in a very very efficient manner. So, what should be in good algorithm? So, let me take this figure; this example to give you the intuition on what kind of algorithm would be used for finding the actual tag sequence, to do it efficiently. So, the idea here is that if you are enumerated, all the 32 possibilities you are doing some computations again and again. So, can you try to reduce the time by doing it in a dynamic programming approach where you first store computations for the smaller paths and use that for the larger paths? So, we have seen one particular example of dining programming in the case of added distance.

We will try to use a similar concept here for decoding the actual tag sequence and this will be called Viterbi decoding. So, looking at the picture, so how you can save the computation time?

(Refer Slide Time: 05:42)



Let us take this sentence here, now let us; let me take 2 different paths. So, one path that is there in the actual sequence S VBD TO VB DT and NN and let me take another path that is VBD TO then JJ then DT and then NN. So, there are 2 paths that we start with the same tag and end with the same tag. So, in VBD TO VB NNP sorry, DT NN and VBD TO JJ DT NN, So now, they are 2 different tag sequences and I want to find out probability of both of these to find out which one is better. Now what is the idea of Viterbi decoding? The idea is that. So, you are ending with the same sequence here DT and NM. So, let us look at only till that point till DT because after DT the computation will be the probabilities will be the same.

This is one way of saving computation in both of these the composition of DT will be the same. So, you can save on that, but what is more important here. So, when you arrive at the tech DT, in one case you are coming from VB, another case you are coming from JJ. So, there are 2 different ways you are coming to DT in these 2 sequences, let us suppose that at this point, I can store for the word or further tech DT, what is the best way of arriving at DT is it via this path or is it via this path. So, I suppose I can find out at this point what is the best way of arriving a DT is it this path of that path.

If I can store, if I can somehow find that the later computations will, so for the 2 paths the later computations will be the same. So, then I do not need to compute these probabilities again for the next sequence. So, if I can store at this point that is the best way is coming from VB and not from JJ. So, then I am doing it for the next steps that are going by DT, I do not

need to explore this path. So, this is the idea suppose I write them as steps, step 1, 2, 3, 4 and 5 at each step for each part of speech tag if I can store, what is the best previous tag from which it is it should come? So, which previous tag will maximize the probability of arriving at this particular tag, then the computation of shared for the same path and I do not need to repeat the same computation again and again.

So, in this particular example, once I can store at step 4 for DT, the best way is coming from VB and not from JJ when I go further, I need not worry about this path at all, I can simply continue with this path because if this probability is P 1, this is P 2, the next probability will be shared, it will be times probability of NN given DT and probability of bill given NN that will be shared across the 2 paths. So, if I know at this point P 2 is higher than P 1, I can probably I can forget this path for going via DT and this is the idea this I will do for each of the speech step.

Similarly for step 3 for each of the 4 part; 4 part of speech tags like for VB I will. So here it is easy because for each part of speech tag it can come only via TO. So, it is important for this case because for DT it can come via 4 previous tags. So, I needed store which of the previous tags it is coming from which has the highest probability similarly fine and P which of the 4 previous tags has the highest probability. So, that is what I am going to do at this step for each part of speech take I will store what is the best probability of arriving at this part of speech.

(Refer Slide Time: 11:14)

*Finding the best path: Viterbi Algorithm*

*Intuition*

Optimal path for each state can be recorded. We need

- Cheapest cost to state  $j$  at step  $s$ :  $\delta_j(s)$
- Backtrace from that state to best predecessor  $\psi_j(s)$

*Computing these values*

- $\delta_j(s+1) = \max_{1 \leq i \leq N} \delta_i(s) p(t_j|t_i) p(w_{s+1}|t_j)$
- $\psi_j(s+1) = \arg\max_{1 \leq i \leq N} \delta_i(s) p(t_j|t_i) p(w_{s+1}|t_j)$

Best final state is  $\arg\max_{1 \leq i \leq N} \delta_i(|S|)$ , we can backtrack from there

Let us look at the formulation, so intuition is that so as I have explained, we are recording what is the optimal path at each state for a given step, what we are storing, what is the cheapest cost of arriving at a particular state; that means, that has the highest probability in the cheapest cost. Now once we have found out what are the tags at different points we also want to write down the sequence. So, I also need to store batteries. So, this is the highest probability. So, what is the particular part of speech tag that gives the highest, this highest probability? So, I want to store this  $\delta_j(S)$  what that is the cheapest cost or the highest probability to step  $j$  at step  $S$  and also the back trace from that state to the best predecessor. So, for the previous example that we were seen for DT the best predecessor was VB and not  $j$ .

Now this I am doing at any step as for state  $j$  how do I use that to compute for the next step? So, let us again look at it. So, suppose I have stored  $\delta_j(S)$ . So, here  $S$  is 3 and  $j$  is my state, it can be VB JJ NN and RB. So, I have stored for all these part 4 different part of speech tags what is the cheapest cost of arriving at this point, what does that mean? I know at this point which path is better VBD TO VB or VB NT or VB. So, I know that for all these 4 different part of speech tags. So, I store the step for this step for this state what is the optimal cost other what is the most probable path.

So, this cost I have or this probability I have now how do I use this probability is to compute for the next step say want to compute  $\delta_{j+1}(S)$ , how will I compute that? So, now,  $\delta_j(S)$  would have taken care of the word back already. So, what are the things that I added for going from? I am sorry; here this should be stepped yes. So, this is  $\delta_j(S) + \delta_{j+1}(S)$ . So, this is a step  $S$  plus 1 I want to find out probability  $\delta_{j+1}(S)$ .

$$\delta_j(s+1) = \max_{1 \leq i \leq N} \delta_i(s) p(t_j | t_i) p(w_{s+1} | t_j)$$

I know the probabilities are step  $S$ , now what is the additional information the transition from this is state to this state and the emission probability of this word given is the state. So, I can compute it using  $\delta_j(S) + \delta_{j+1}(S)$  is  $\delta_i(S)$  times the probability of this tag sequence that is probability of tag  $t_j$  given  $t_i$  this is the probability of tag sequence transition and the emission probability that is probability of word edge step  $S$  plus 1 given  $t_j$ .

)

$$\psi_j(s+1) = \operatorname{argmax}_{1 \leq i \leq N} \delta_i(s) p(t_j | t_i) p(w_{s+1} | t_j)$$

This I am doing for some  $i$ th state, but there are 4  $P$ , there can be some  $N$  Number of states. So, I will find out the max of those among all the states at the previous step I have  $\delta_S$   $i$  multiplied with it the transition probability and the emission probability at step  $S$  plus 1 and the max gives me what is the best path at this is state  $j$  at step as for this 1. So, now, suppose out of these 4, I found out this is the best path. So, we will also like to store what is the best predecessor and this is nothing, but argmax of this function. So, I call that size  $S$  plus one that is argmax of the same function and that is what I am also storing and that is all this will continue. So, what will happen at the end I am at a step 5 and I know, what is the probability? So, let me write down  $\delta_{NN}$  at 5 and  $\delta_{VB}$  at 5, I know both these probabilities, what is the best way of reaching at this state, what is the best probability of reaching at this state at this fine I can just take the maximum of these 2 and this will give me the best probability of any possible part of speech tag sequence and then I can use the back traces from here to obtain the actual part of speech tag sequence and this is my vita by decoding algorithm.

So, if we see that formally, so yes, so I am storing at each step  $S$ , what is the cheapest cause of the best probability of reaching there also the predecessor and using that I am also computing the probabilities for the next step that is. So, that is similar to what we have seen here the probability at the previous step transition probability and emission probability then I take the max over all the previous possible states also show the predecessor for the next step and how do we end once we have gone through the sentence end I will have the probabilities for all the states possible at the end the ending word I will take the maximum of those n back trace from there and that will give me the optimal automat tag sequence for this particular sentence.

(Refer Slide Time: 18:26)

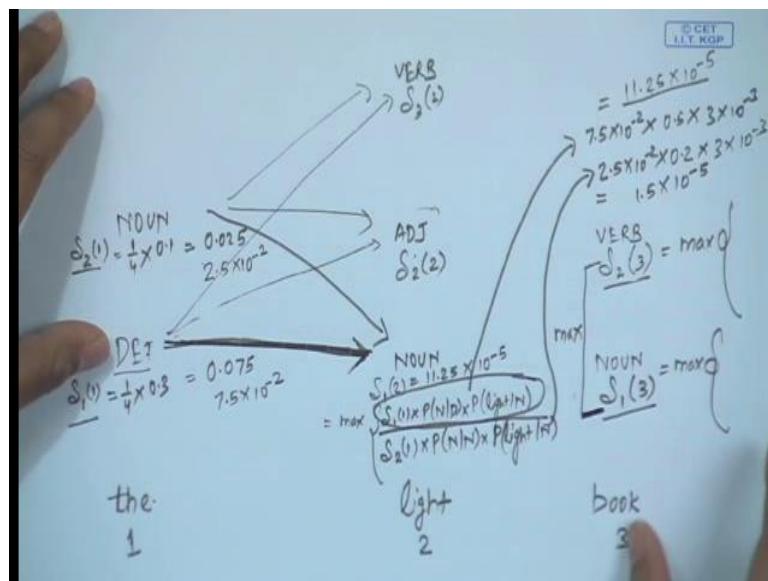
**Practice Question**

- Suppose you want to use a HMM tagger to tag the phrase, "the light book", where we have the following probabilities:
- $P(\text{the}|\text{Det}) = 0.3, P(\text{the}|\text{Noun}) = 0.1, P(\text{light}|\text{Noun}) = 0.003, P(\text{light}|\text{Adj}) = 0.002, P(\text{light}|\text{Verb}) = 0.06, P(\text{book}|\text{Noun}) = 0.003, P(\text{book}|\text{Verb}) = 0.01$
- $P(\text{Verb}|\text{Det}) = 0.00001, P(\text{Noun}|\text{Det}) = 0.5, P(\text{Adj}|\text{Det}) = 0.3, P(\text{Noun}|\text{Noun}) = 0.2, P(\text{Adj}|\text{Noun}) = 0.002, P(\text{Noun}|\text{Adj}) = 0.2, P(\text{Noun}|\text{Verb}) = 0.3, P(\text{Verb}|\text{Noun}) = 0.3, P(\text{Verb}|\text{Adj}) = 0.001, P(\text{Verb}|\text{Verb}) = 0.1$
- Work out in details the steps of the Viterbi algorithm. You can use a Table to show the steps. Assume all other conditional probabilities, not mentioned to be zero. Also, assume that all tags have the same probabilities to appear in the beginning of a sentence.

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 1      5/6

I hope this algorithm is clear. So, let us take a simple example. So, that you can become much more familiar with that how do you use this video decoding algorithm finding problem. So, what do you see?

(Refer Slide Time: 18:47)



In this problem I have a sentence the light book and you are given some sort of emission probabilities first that the word the can come from the tag determiner or noun. So, they are 2 possibilities determiner noun the word light can come from noun adjective and verb possibility by using vita b decoded algorithm. So, how do we start? So, I have 3 steps 1 2 3.

So, I want to store at each step for each state. So, let me call it delta one at step one similarly delta 2 as step one. So, what is this probability? So, this will be what is the probability that the tag determiner occurs in the start of the sentence multiply with the probability of emitting the from the tag determiner.

Let us see how these probabilities given in this case probability of determine occurring in the start of the sentence. So, in this question in the last sentence, it is written assume that all the tags that have the same probability stop in the beginning of the sentence. So, probabilities are given implicitly. So, now, the question is it how many tags would you assume. So, here you can take any assumption. So, in this question you are shown only 4 tags determiner noun verb an adjective. So, assume there are only 4 part of speech tags and each one has equal probability of occurring at the start of the sentence. So, what will the probability of determine occurring at the start of sentence it will be one by 4 similarly for noun one by 4 times probability of the given determiner and that is given here h point 3 I am probability of the given noun is given as point 1. So, this is your delta 2 1 and delta 1 1. So, you can also further write them oh 0.025 and this will be 0.075. So, better write them as  $2.5 \times 10^{-2}$  minus  $2.5 \times 10^{-2}$ .

You have delta 1 1, del 2 1, now you go to the next step. So, now, here you want to find out delta 1 2 in step 2, what is the best probability of reaching noun? You can reach via either here or here. So, you need to compute both the possibilities and take the max over that. So, let us compute these probabilities. So, what will be that so, first will be delta 1 1 times probability of getting noun from determine times probability of light given noun S and second one will be delta 2 1 times probability noun given noun times probability light given noun. So, let us compute these 2 probabilities. So, you will take the max of these.

So, let us compute these probabilities. So, this would be. So, let me take it here seven point 5 into 10 to the power minus 2 probability of noun given to determiner. So, that is that we can see from the table given to us noun given determiner point 5 and light given noun each 3 into 10 to power minus 3 and this gives me  $3.75 \times 10^{-3}$  11.25 into 10 to the minus 5 and the second one becomes  $2.5 \times 10^{-2}$  times probability of n; given n, this is 0.2 into probably at given and that will be 3 into 10 to the minus 3 and that is 0.5 1.5; 1.5 into 10 to power minus 5 and from these 2 you can say that this one is higher than the next one. So, this is what you will take. So, you can store delta 1 2 H 11.25 into 10 to power minus 5 and you will also store the back trace that is coming from this path same thing you will do for

adjective and verb adjective again there will be 2 possibilities coming from determiner and noun for work there will be 2 possibilities again you compute the probabilities using the formula and store what is the best probability. So, it will be delta 3 at step 2 delta 2 h step.

This is what you will store; also you will store the back trace, once you have done that now we will go to the final step. So, where you will store delta 1 step 3 1 means noun here and delta 2 at the step 3 again you will compute that by taking all the 3 possibilities times the transition and all. So, this will be max over 3 quantity this will be max over 3 quantities again you will find out the 2 values delta one 3 and delta 2 3 and how will you actually determine the finest sequence you will take the max of this suppose max comes out to be this one then you will go to the predecessor suppose the predecessor was this one you will go to hear suppose predecessor of as this one and so on. So, you will find out this sequence assuming this is these are the best fervent.

But I hope the idea is clear that how do you actually do this computation? So, I would encourage that you complete this particular exercise on your own.

(Refer Slide Time: 27:26)

*Learning the Parameters*

*Two Scenarios*

- A labeled dataset is available, with the POS category of individual words in a corpus
- Only the corpus is available, but not labeled with the POS categories

*Methods for these scenarios*

- For the first scenario, parameters can be directly estimated using maximum likelihood estimate from the labeled dataset
- For the second scenario, *Baum-Welch Algorithm* is used to estimate the parameters of the hidden markov model.

Ptwan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 1      6 / 6

Now, coming to the point of learning the parameters so, what are the parameters do we need in the; so, we need 3 different parameters. So, we need the probability of starting the sentence.

(Refer Slide Time: 27:50)

$$A = \{a_{ij}\}_{\text{all pos tags}}$$

$$B = \{p(\text{word}|t_i)\}_{\text{all pos tags}}$$

$$\xrightarrow{\text{parameters of the HMM}}$$

S1. Labeled Dataset  $p(t_j|t_i) = \frac{c(t_i, t_j)}{L(t_i)}$  MLE

S2. Corpus but no labels  $p(t_j|t_i)$

That is let me call this in the language of HMM the pi that is what is the probability that a particular tag or estate start the sequence. So, I want this probability  $\pi_i$  for all pos tags then I need the probability of transition from one part of speech at another. So, this is my transition matrix I need all this  $a_{ij}$  transiting from probability  $T_j$  given  $T_i$  and I need my emission probabilities that is probability of the word given a tag. So, you can use it using a matrix  $b$ . So, I need all these 3 probabilities to actually use this Viterbi decoding algorithm at runtime if I do not have these probabilities I cannot teach you Viterbi decoding.

The question is so; these are my parameters of the HM. So, how do we actually find out these parameters of my HM? So, here I am saying there are 2 different scenarios that you might have. So, first scenario is where a label data set is available what do you mean by a label data set you have a set of sentences available to you for each sentence you also know what is the actual part of speech tags sequence. So, what is the part of speech tag for individual word somebody is manually labeled it for you. So, this is a label data set this one scenario second scenario is you only have the corpus, but you do not have any data where the sentences are labeled with the part of speech categories. So, I have the tools that is a clear. So, scenario one where you have a label data set a scenario to only corpus, but no labels. So, what I am saying for these 2 scenarios you can find the parameters in different manner. So, suppose you have scenario 1 you have a label data set. So, what you can do for finding the parameters of HMM. So, this is something we also took them in the last week.

For example, if the label data set, I want to find a parameter like them the transition probabilities probability of tags a given tag I, how will you find that the label data set you will have all the possible tag sequences that actually occurred in the corpus now from that you will find out number of times  $t_i$  n  $t_j$  occur together in the corpus divided by number of times only  $t_i$  occurs. So, this was what we said as maximum likelihood estimator you can use that to find this probability same with all the other probabilities  $p_i$  as well as  $P$  you can find using maximum likelihood estimate.

Now, problem comes when you have the corpus, but no labels. So, now, there are no labels. So, how would you actually find out probability of  $t_j$  given  $t_i$  when in the data there is no label on which word got a tag  $t_i$  or which one got it at  $t_j$ . So, you cannot compute it directly from any labels. So, what is the method that you will use for learning the parameters when the corpus is available, but labels are not available and that is what we will see in the next lecture.

We will be using Baum Welch algorithm to estimate the parameters of the hidden Markov model when the labels are not available. So, this is similar to expectation maximization algorithm and you will see what is the addition for this algorithm and how we actually apply this. So, hope in this lecture you understood how to apply with Viterbi decoding when the parameters of HMM are given and so you can do that for (Refer Time: 32:32) and in the next lecture we will discuss how do we run the parameters when the labels are not available.

Thank you.

**National Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 18**  
**Baum Welch Algorithm**

So, welcome back for the second lecture of this week. So, in the last lecture, we were discussing Viterbi decoding for HMMs. And in the end, we discussed the problem of learning the parameters of HMMs. And we say that when the label data set available, we can simply estimate using maximum likelihood estimate by the label data set. But if the label data set is not available then how do we actually learn the parameters of the system and we said we will be using some sort of expectation algorithm and in this particular case this is called Baum Welch algorithm.

(Refer Slide Time: 00:57)

**Baum Welch Algorithm**

Uses the well-known EM algorithm to find the maximum likelihood estimate of the parameters of a hidden markov model

**Parameters of HMM**

Let  $X_t$  be the random variable denoting hidden state at time  $t$ , and  $Y_t$  be the observation variable at time  $T$ . HMM parameters are given by  $\theta = (A, B, \pi)$  where

- $A = \{a_{ij}\} = P(X_t = j | X_{t-1} = i)$  is the state transition matrix
- $\pi = \{\pi_i\} = P(X_1 = i)$  is the initial state distribution
- $B = \{b_j(y_t)\} = P(Y_t = y_t | X_t = j)$  is the emission matrix

Given observation sequences  $Y = (Y_1 = y_1, Y_2 = y_2, \dots, Y_T = y_T)$ , the algorithm tries to find the parameters  $\theta$  that maximise the probability of the observation.

Pawan Goyal (IIT Kharagpur) POS Tagging Week 4, Lecture 2 2/6

So, Baum Welch algorithm as I was saying this is a well known EM algorithm to estimate the maximum likelihood for the parameters of the HMM. So, what are the parameters of HMM that you want to estimate let we just formally define them and then we will see how do we estimate them using Baum Welch. So, in the HMM model, so we have a hidden state at every time point and an observation variable. So, here I am using capital X t to denote the random variable that is the hidden state and capital Y to denote the observation variable. So, both X t and Y t can take many different values; for our part of speech tag is each X t can take one of

the part of speech tags, and each  $Y_t$  can take one of the possible words. So, this my hidden variable hidden state  $X_t$  and the observation  $Y_t$ .

The parameters are  $A$ ,  $B$  and  $\pi$  that we discussed last time. What is  $A$ ? So  $A$  is my state transition matrix. So, given that the previous state was  $i$  what is the probability that the current state will be  $j$ ? So, here the  $t$  as such does not matter what matters is that the state  $j$  occurred after state  $i$  at any given time point. Then I have the parameter  $\pi$  that is what is the probability that this particular state is starts the sequence. So, probability of  $X_1$  that the first state being the  $i$  x state that is my  $\pi_i$ . And then thirdly I have the emission matrix. So, where the entries are what is the probability of observing this particular word or this observation given the current state. So, this I am denoting like  $b_j(y_t)$ ; at the state  $j$  what is the probability of emitting the observation variable or observation  $y$ . So, either three parameters three set of parameters that I have learnt.

Now, what is given to me? So, I was saying that we are given the corpus not labeled, but the word sequences are given. So, this can be seen as my observations are given different sentences are given, I knew what are different words occurred in the sentences. So, I can say that I am given a set of observations in data. So, here I can say that I am given a set of observation sequences that the first observation was the word  $Y_1$  then  $Y_2$  up to some  $Y_t$ , where  $Y_t$  denote the end of the sentence. So, I am given a set of observation sequences and my aim is to find out the optimal set of parameters  $\theta$  that maximize the probability of this observation. So, these observation sequences, so that is where I am using Baum Welch algorithm, find out what are the optimal set of parameters  $\theta$  that will maximize the probability of likelihood of observing this observation that is why I am using expectation maximization algorithm.

(Refer Slide Time: 04:31)

The basic idea is to start with some random initial conditions on the parameters  $\theta$ , estimate best values of state paths  $X_t$  using these, then re-estimate the parameters  $\theta$  using the just-computed values of  $X_t$ , iteratively.

*Intuition*

- Choose some initial values for  $\theta = (A, B, \pi)$ .
- Repeat the following step until convergence:
- Determine probable (state) paths  $\dots X_{t-1} = i, X_t = j \dots$
- Count the expected number of transitions  $a_{ij}$  as well as the expected number of times, various emissions  $b_j(y_t)$  are made
- Re-estimate  $\theta = (A, B, \pi)$  using  $a_{ij}$  and  $b_j(y_t)$ s.

A forward-backward algorithm is used for finding probable paths.

Pawan Goyal (IIT Kharagpur) POS Tagging Week 4, Lecture 2 3 / 6

So, what is the main intuition of EM? So, idea would be I want to estimate the parameters A, B, and  $\pi$ . So, what I will do? I will start with some random initial probabilities for all these parameters; I will initialize them with some random values. So, I have all these probabilities. Now, using these initial values, I will try to find out what are the probabilities of various state paths. So, on a given sequence, what is the probability that a particular state would have occurred given these parameters? So, once I have obtained the probabilities of different state paths or different states at a different point, now I got this probability, I will use that again to compute to re-estimate my parameters. So, I will get started with theta zero, I get some theta prime. Now, I will again use the theta prime to complete my state paths probabilities again use that to re-compute my parameters until it somehow converges.

So, this is my iterative algorithm. First use some parameters theta to get some likely from the data for the hidden variable here that is the states. Once you have that likelihood or the probabilities use that to compute my theta. So, what we are doing here? So, I am starting by choosing some initial values of the parameters A, B, and  $\pi$ . And then I have to repeat the following steps until convergence. What is that? Firstly, determine what are the probable paths that  $X_{t-1}$  that  $t-1$  object point I see the state  $i$ ; at  $t$ -th, I see the state  $j$  and. So, on what is the probability of various state paths.

Now, once I have these will teach count the number of times. So, what is important expect a number of times, because we are only computing probabilities, we are not finding the actual

paths. So, count the expected number of transitions  $a_{ij}$  as well as the expected number of times various emissions  $b_j(y_t)$  occur. So, using these state paths can you do that, yes, I know what is the probability of  $X_t$  minus 1 is equal to  $i$ ;  $X_t$  is equal to  $j$ ; I can use it to compute an expected value for  $a_{ij}$ , and this is what we will be doing. So, we will compute  $a_{ij}$ , similarly I already know the observation. So, I can also compute the expected vanish for  $b_j(y_t)$  what is the probability that I see objection  $y_t$  for the particular state  $j$ . So, once I computed this  $a_{ij}$  and  $b_j(y_t)$ , I will again estimate my parameters theta using these computed values.

Now, I will go back in the loop, I have this theta, I will compute the probabilities of various state paths again compute  $a_{ij}$ ,  $b_j(y_t)$  and again compute theta and I repeat that until convergence. So, now so all these are interesting here like, firstly, once you are given some set of parameters theta it can be either the initial parameters or some intermediate parameters how do you actually compute different the probabilities of different paths, how do you actually do it. And for that we use a forward backward algorithm that is the main concept of this algorithm.

(Refer Slide Time: 08:26)

### Forward-Backward Algorithm

*Forward Procedure*

$\alpha_i(t) = P(Y_1 = y_1, \dots, Y_t = y_t, X_t = i | \theta)$  be the probability of seeing  $y_1, \dots, y_t$  and being in state  $i$  at time  $t$ . Found recursively using:

- $\alpha_i(1) = \pi_i b_i(y_1)$
- $\alpha_j(t+1) = b_j(y_{t+1}) \sum_{i=1}^N \alpha_i(t) a_{ij}$

---

*Backward Procedure*

$\beta_i(t) = P(Y_{t+1} = y_{t+1}, \dots, Y_T = y_T | X_t = i, \theta)$  be the probability of ending partial sequence  $y_{t+1}, \dots, y_T$  given starting state  $i$  at time  $t$ .  $\beta_i(t)$  is computed recursively as:

- $\beta_i(T) = 1$
- $\beta_i(t) = \sum_{j=1}^N \beta_j(t+1) a_{ij} b_j(y_{t+1})$

Pawan Goyal (IIT Kharagpur) POS Tagging Week 4, Lecture 2 / 476

So, let us see what is this forward backward algorithm. So, in forward-backward algorithm, we have a forward procedure and a backup procedure. So, so far for explaining let me just take show it on paper once.

(Refer Slide Time: 08:47)

$$P(X_t=i | Y, \theta) \quad P(X_t=i, X_{t+1}=j | Y, \theta)$$

$$\alpha_i(t) = P(Y_1=y_1, Y_2=y_2, \dots, Y_t=y_t, X_t=i | \theta)$$

$$\beta_i(t) = P(Y_{t+1}=y_{t+1}, \dots, Y_T=y_T | X_t=i, \theta)$$

$$\alpha_i(t+1) = \sum_{j=1}^N \alpha_j(t) a_{ji} b_j(y_{t+1})$$

So, what is happening? I am observing this sequence  $Y_1, Y_2, \dots, Y_{T-1}, Y_T$ , this is my observation. And correspondingly, so you can say this is  $Y_1$  is equal to  $y_1$  first observation,  $Y_2$  is equal to  $y_2$  and so on. And correspondingly there are states, so this is my observation and then there are states. So, there will be some state  $x_t$  this is  $y_t$  observation, this is my state  $x_t$  is equal to  $y_t$  it can be any of the possible states  $x_t$ . Now, what do I do in this forward backward algorithm, I compute two different probabilities; one I called as the forward probability. This is the probability of observing  $y_1$  to  $y_t$  plus and  $x_t$  is equal to  $i$  give my parameter  $\theta$ . So, this is my  $\alpha_t$  for the  $i$ th the state. The probability that I am observing  $y_1$  to  $y_t$  and at  $t$ -th point the state is  $i$  given my parameters  $\theta$ ;  $Y_1$  is equal to  $y_1$ ,  $Y_2$  is equal to  $y_2$ ,  $Y_t$  is equal to  $y_t$  and  $x_t$  is equal to  $i$  given parameters  $\theta$ . This is my forward probability.

Then I compute a backward probability. What is that? This is the probability of observing this given this state. So, this is my  $\alpha_t$ , and this is  $\beta_t$  that is probability that  $Y_{t+1}$  is small  $y_{t+1}$  up to capital  $Y_T$  is  $y_T$  given  $X_t$  is equal to  $i$  and  $\theta$ . So, these are two different probabilities that I stored. So, what do you see here, so I am taking just a state at a particular time  $t$  a particular state  $i$ . Now, why do you actually compute this  $\alpha_t$  and  $\beta_t$ , how does that help? Now I have these probabilities, I can simply multiply these  $\alpha_t$  times  $\beta_t$  to get the probability that I observe the sequence and the  $i$ th the state. And to actually compute the probability of saying that at  $t$ th time point the state was  $i$ , I can marginalize it by all the possible states that can happen at time  $t$ . So, I can multiply  $\sigma_i t$ ,

beta  $i t$  and divide by all the possible sigma  $j t$  alpha  $j t$  and beta  $j t$  to compute the probability of seeing state  $i$  at time point  $t$  and this is that is how this forward and backward probabilities help. So, you will see that in detail.

So, right now just the formulation. So, I am this higher the sequence, I divided in two parts for a given  $t$ . This is state  $i$ , I have the forward probability alpha  $i t$  and a backward probability beta  $i t$ ; alpha  $i t$  is this particular sequence and the state, beta  $i t$  is the ending sequence given this is state; the previous state  $X t$  is equal to  $i$ . So, now so this is my alpha  $i t$ . So, now, how do I compute these values of alpha  $i t$ ? I want to compute this for all possible  $i$ 's and all possible values of  $t$ . So, this is a forward probability.

So, I have to start with the first point. So, how do I actually compute alpha  $i 1$ , what is that? So, alpha  $i 1$  from this equation which probability of observing  $Y 1$  is equal to  $y 1$  and  $X 1$  is  $i$  given theta. So, what is the probability of  $X 1$  is equal to  $i$  given theta that is  $\pi i$ ; this is my  $\pi i$ , and this probability would be emission from  $X 1$  from the either state. So, this will be  $b i y 1$ . So, this is how I will compute the initial one alpha  $i 1$  for all possible states  $i$ . Now, at some point of time, at some point  $t$  plus 1, how will I compute alpha  $j t$  plus one given the previous alpha  $i t$ , I have computed. So, alpha  $j t$  plus 1, so  $t$  plus 1 means the probability of seeing the sequence up to  $Y t$  plus 1 and the state  $X t$  plus 1 is  $j$ .

So, how can I compute that using the previous alphas? So, I have the previous alpha that gives me this. So, alpha  $i t$  at the previous time step I had some state, I have the transition from  $i$  to  $j$ , so times  $a i j$  and now I have the emission for the  $t$  plus 1th observation that is  $b j y t$  plus 1. And this I can obtain from any of the previous states. So, I have to sum over all the possible states. So, from all the previous states at the  $t$ -th time step, I can be any state  $i$ , I have the transition probabilities and the emission probability for  $t$  plus 1 and that is how I can compute sigma alpha  $j t$  plus 1 from all the alpha  $i t$ -th. So, this is how you can compute your alpha in a recursive manner starting from alpha  $i 1$ .

Now what will you do for your betas. So, this is a backward procedure. So, what is my beta  $i t$  this gives me the probability of ending the sequence with  $y t$  plus 1 to  $y t$  given the state  $i$  at time, this is my backward procedure. Now, this I again compute recursively in alpha  $i$  was starting from the first word I was computing alpha  $i t$ . In betas because the backward procedure, I will first compute beta  $i T$ ,  $T$  is the end of the sequence and what will the beta  $i t$ , so beta  $i t$  would be probability of so given that previous the  $t$ -th point state is  $i$ .

having a sequence on  $t + 1$ , but there is no word from  $t + 1$ . So, this will always be. So, this is always be 1, this will be a null sequence. So, this is always one. Now, I have to compute it recursively for all other beta  $i$  at time point  $t$ . So, here I am going backwards.

So, you assume that you have computed all the beta  $j$  at time  $t + 1$  and you want to use that to compute beta  $i$  at time point  $t$ , how will you do that? So, this is very similar to what you did in the case of alpha. So, we can see this. So, this is beta  $j$  at time  $t + 1$  transition probabilities from  $i$  to  $j$  and the emission probabilities probability for the  $t + 1$ th observation, and you sum over all the possible states at times  $j$  so very, very similar to what you did in the case of forward procedure. So, that is what you will do in the backward procedure for computing beta  $h$  in a recursive manner. So, now, you have a way that given the set of parameters minus theta you can compute all this alpha and beta. So, all the alpha  $i$  at  $t$  is and beta  $i$  at  $t$  you can compute.

Now, remember what is the next step in the algorithm, you want to find out what are the best possible paths or what is the probability of various paths that is what we intended by doing this alpha and beta, let us see how exactly we can compute these probabilities of various paths. So, I want to compute the probabilities of say probability  $X$  is equal to  $y$  given my observation and parameters theta, also I want to compute  $X_t$  is equal to  $y$  and  $X_{t+1}$  is equal to  $j$  given  $y$  and theta, I want to compute both of this. So, if I can compute both of this, I can compute all the parameters, I can complete my transition probabilities using that, I can compute my initial probabilities using that by taking  $t$  is equal to 1 and you can also compute my ambition probabilities. So, now, my problem is once I computed alphas and betas can I compute these probabilities, so that is what we will see next.

(Refer Slide Time: 19:42)

*Finding probabilities of paths*

We compute the following variables:

- Probability of being in state  $i$  at time  $t$  given the observation  $Y$  and parameters  $\theta$

$$\gamma_i(t) = \frac{P(X_t = i | Y, \theta)}{\sum_{j=1}^N \alpha_j(t) \beta_j(t)} = \frac{a_i(t) \beta_i(t)}{\sum_{j=1}^N a_j(t) \beta_j(t)} = \frac{P(X_t = i, Y | \theta)}{P(Y | \theta)}$$

- Probability of being in state  $i$  and  $j$  at time  $t$  and  $t+1$  respectively given the observation  $Y$  and parameters  $\theta$

$$\zeta_{ij}(t) = P(X_t = i, X_{t+1} = j | Y, \theta) = \frac{a_i(t) a_j(t+1) b_j(y_{t+1})}{\sum_{i=1}^N \sum_{j=1}^N a_i(t) a_j(t+1) b_j(y_{t+1})} = \frac{P(X_t = i, X_{t+1} = j, Y | \theta)}{\sum_{i=1}^N \sum_{j=1}^N P(X_t = i, X_{t+1} = j, Y | \theta)}$$

$$\gamma_i(t) = P(X_t = i | Y, \theta) = \alpha_i(t) \beta_i(t) / \sum_{j=1}^N \alpha_j(t) \beta_j(t)$$

So, I am giving these some names. So, firstly, I have this gamma  $i t$  that gives me the probability that at  $t$ -th point the state is  $i$  given my observations  $y$  and theta parameters. And similarly, I have zeta  $i j t$  that is probability that at  $t$ -th point, I have state  $i$ ;  $t$  plus 1 point, I have state  $j$  given  $y$  and theta. So, gamma  $i t$  is written terms of multiplication of alpha  $i t$  and beta  $i t$  divided by summation over all possible sigma  $z$  alpha  $j t$  and beta  $j t$  over all possible change. So, now how do we actually come up with this equation of gamma  $i t$  and zeta  $i j t$ , let us look at it.

So, here I want to compute probability  $x t$  is equal to  $y$  given  $y n$  theta. So, now, I will computed using probability  $X t$  is equal to  $y$ ,  $Y$  given theta divided by probability  $Y$  given theta, this is simple conditions. So, if you forgot that theta, this is nothing but  $X t$  is equal to  $i$  given  $Y$ ,  $X t$  is equal to  $y$ ,  $Y$  divided by probability  $y$ , this simple the condition probability rule. Now, this one I am writing as summation  $j$  is equal to 1 over  $N$ ,  $j$  is equal to 1 to  $N$  probability  $X t$  is equal to  $j$ ,  $y$  given theta. So, now, you can see the symmetry in this equation.

Now, what I have to show that alpha  $i t$  times beta  $i t$  is probability  $X t$  is equal to  $y$ ,  $Y$  given theta and that is very easy, because what is alpha  $i t$  this is probability  $y 1$  to  $y t$  plus  $x t$  is

equal to  $i$ . What is beta  $i t$  probability of  $y t$  plus 1 to  $y$  capital  $T$  and now  $X t$  is equal to  $y$  is already given for beta, and this is given my parameters theta and this is what we are seeing here. This is my alpha  $i t$  times beta  $i t$  and this is marginalizing over all possible states are time by  $t$ . So, this will be sigma,  $j$  is equal to 1 to  $N$ , alpha  $j t$  beta  $j t$ . So, this gives me the equation for gamma  $i t$ .

Similarly, you can see for zeta  $i j t$  that is for probability of  $X t$  is equal to  $i$   $X t$  plus 1 is equal to  $j$   $Y$  given  $Y n$  theta. So, in the same way, you can write it as probability  $X t$  is equal to  $i$   $X t$  plus 1 is equal to  $j$ ,  $Y$  given theta divided by probability  $Y$  given theta that will marginalize over all possible  $i$  and  $j$ . And you will say probability  $X t$  is equal to  $i x t$  plus 1 is equal to  $j y$  given theta for all possible values of phi and  $j$ .

Now, what we have to show that this is actually this formula now. So, what is this? So, you are given state at time point  $t$  and time point  $t$  plus 1. So, what you are actually doing is that using the forward procedure from  $y 1$  to  $y t$ , and you will also get the state  $i$  at time point  $t$ . And you will use the backward procedure because you are given the state at time  $t$  plus 1 state  $j$ , backward procedure for from  $y t$  plus 2 up to  $y$  capital  $T$ . So, this is what is captured in alpha  $i t$ , and this is captured condition on  $j$  in beta  $j t$  plus 1. Then you will compute the probability of transition from  $i$  to  $j$  that is  $a i j$  and  $y t$  plus 1 given  $j$  that is your  $b j y t$  plus 1. So, this equation is nothing but probability of  $X t$  is equal to  $i$ ,  $X t$  plus 1 is equal to  $j$   $Y$  given theta, this whole sequence is there and these two states are also there. And similarly you can see for here this is over all the possible values of  $i$  and  $j$ .

So, I hope this is clear that once I have computed all the possible alpha and beta and the previous parameters of there, we can compute this gamma  $i t$  and zeta  $i j t$ -th that are probability of state at a given time and a sequence  $i$  and  $j$  at time  $t$  and  $t$  plus 1, this we can compute using alpha and beta. Now, coming to the last part that is once I computed some probability is over by possible state paths how do I estimate my parameters again, I have started with some initial parameters computed this, now I want to re-estimate my parameters theta. So, how do I do that?

(Refer Slide Time: 26:14)

### Updating the parameters

- $\pi_i = \gamma_i(1)$ , expected number of times state  $i$  was seen at time 1
- $a_{ij} = \frac{\sum_{t=1}^T \zeta_{ij}(t)}{\sum_{t=1}^T \gamma_i(t)}$ , expected number of transitions from state  $i$  to state  $j$ , compared to the total number of transitions away from state  $i$
- $b_i(v_k) = \frac{\sum_{t=1}^T \mathbf{1}_{y_t=v_k} \gamma_i(t)}{\sum_{t=1}^T \gamma_i(t)}$  with  $\mathbf{1}_{y_t=v_k}$  being an indicator function, is the expected number of times the output observations are  $v_k$  while being in state  $i$  compared to the expected total number of times in state  $i$ .

Pawan Goyal (IIT Kharagpur) POS Tagging Week 4, Lecture 2 6/6

So, now I have to estimate my  $\pi_i$ ,  $a_{ij}$  and  $b_i(v_k)$ . So, how do I estimate this  $\pi_i$  is probability of seeing the state  $i$  at time one. So, now, from these parameters which one can give me this value probability that  $x_1$  is equal to  $i$ ; this I can get from  $\gamma_i(1)$  that is that can give me my  $\pi_i$  is probability that at  $X_1$  is  $i$ . So, this is to estimate my  $\pi_i$ . Next thing I have to estimate is transition probabilities probability of going from state  $i$  to state  $j$ . Now I have computed this.

$$b_i(v_k) = (\sum_{t=1}^T \mathbf{1}_{y_t=v_k} \gamma_i(t)) / \sum_{t=1}^T \gamma_i(t)$$

(Refer Slide Time: 27:18)

Updating the parameters

$$\begin{aligned} \zeta_{ij}(t) &= P(X_t = i, X_{t+1} = j | Y, \theta) = \frac{a_i(t) a_j \beta_j(t+1) b_j(y_{t+1})}{\sum_{i=1}^N \sum_{j=1}^N a_i(t) a_j \beta_j(t+1) b_j(y_{t+1})} \\ &= \frac{\sum_j P(X_{t+1} = j, X_t = i, Y | \theta)}{\sum_i \sum_j P(X_{t+1} = j, X_t = i, Y | \theta)} \end{aligned}$$

Pawan Goyal (IIT Kharagpur) POS Tagging Week 4, Lecture 2 5/6

$$a_{ij} = \frac{\sum_t \zeta_{ij}(t)}{\sum_t \gamma_i(t)} \quad b_j(v_k) = \frac{\sum_t \gamma_j(t) \cdot \mathbf{1}_{v_k=y_t}}{\sum_t \gamma_j(t)}$$

$$\mathbf{1}_{v_k=y_t} = \begin{cases} 1 & \text{if } v_k = y_t \\ 0 & \text{otherwise} \end{cases}$$


So, how do I compute my  $a_{ij}$ ,  $a_{ij}$  would be at all possible times how many times you are having the sequence  $i$   $n$   $j$ . So, this will be summation over  $t$   $\zeta_{ijt}$ ,  $\zeta_{ijt}$  is  $i$  followed by  $j$  divide by over all the times when you are seeing this  $t$ . So, when you are seeing the state  $i$  and this you can compute using your  $\gamma_i$  that is summation over  $t$   $\gamma_{it}$ , and that will give me my  $a_{ij}$ . All the possible cases where all the probabilities of  $i$  and  $j$  occurring together at any point divide by all the probabilities of  $i$  occurring anywhere, this is my  $a_{ij}$ .

And similarly, how can you compute your  $b_j$  say  $v_k$  that is at state  $j$  you are emitting  $v_k$  how can you estimate that. So, for that you have to find out whenever you are in state  $j$  is the observation actually  $v_k$ , how many times you are in state  $j$  by the observation is we can divide by number of times you are in state  $j$ . So, this is so here the denominator is easy that is number of times you are, so this  $b_j v_k$  number of times you are in state  $j$  this is  $\gamma_{jt}$  numerator number of times or the probability that you are in state  $j$  times the observation is actually  $v_k$ . Now, so at time  $t$ , the observation is small  $y_t$ . So, I can use some sort of a simple notation or indicator function  $1_{v_k}$  is equal to  $y_t$ , so indicator function. What is indicator function? So,  $1_{v_k}$  is equal to  $y_t$  will be 1 if  $v_k$  is actually  $y_t$  and 0 otherwise this will be my indicator function.

So, now, whenever if  $v_k$  is  $y_t$  whenever I observe  $y_t$ , I will add it, otherwise it will be 0. And this is this will give me the emission probabilities. So, I could estimate my  $\pi_i$ ,  $a_{ij}$  as well as  $b_j v_k$  using this  $\zeta_{ij}$  and  $\gamma_i$  which I computed using my forward backward probabilities. And this is one complete pass of this algorithm. Now, I have these parameters, I will plug these again to compute alphas and betas; again compute my  $\gamma$  and  $\zeta$ s and again estimate the parameters until it converges, and this is the Baum Welch algorithm for learning the parameters of HMM when the label data set is not available. So, this gives you a very nice handle on HMMs.

So, you can apply HMMs, when you have the label data set very easily by learning the parameters directly from memories, maximum likelihood estimate, and data beta decoding. But even if you get a new problem or a new language furtherer you have to compute part-of-speech tag. But you do not have the labels you can use it this Baum Welch algorithm to estimate the parameters of a HMM, and then each bit have de coding to actually given a new sentence find at part of speech tag sequence. So, you can use that for any other sequence label task.

So, this completes our discussions on HMMs that were one of the very popular models for this part of speech tagging in many other sequence labeling task, but they have certain limitations and that is what we will discuss next then that what are these limitations, and how do some other models. So, we will go to now some discriminative classifiers like maximum entropy models they get rid of these limitations. So, this is for HMMs. Let us look at maximum entropy models in the next lecture.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 19**  
**Maximum Entropy Models – I**

Welcome back for the 3rd module of this week. So, in the last week we have; in the last module we had finished the discussions on hidden Markov models how do we learn the parameters and how do we use that for estimating the part of speech tags sequences for a given sentence. Now hidden Markov models for generative models and if you remember we discussed this distinction between generative model and discriminative model. So, we will see a discriminative model; maximum entropy model for this sequence labeling task, but before that let me start with some sort of motivation and why do we move from HMMs to maximum entropy model what are the limitations of HMM.

(Refer Slide Time: 01:00)

*Issues with Markov Model Tagging*

*Unknown Words*  
We do not have the required probabilities.  
*Possible solutions:*

- Use morphological cues (capitalization, suffix) to assign a more calculated guess

*Limited Context*

- "is clearly **marked**" → verb, past participle
- "he clearly **marked**" → verb, past tense

*Possible solution:* Use higher order model, combine various n-gram models to avoid sparseness problem

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lec 19

Then problem with the HMMs also, this is known as the Markov model tagging is handling unknown words suppose at run time, you get some unknown word that was not there in the corpus because the emission probabilities will not be known, there is no where you can come to you can actually use the Viterbi decoding. So, what do you do in that case? So, you might want to use some sort of morphological cues, suppose the word we word is ending with e d, you might think that over this might be a past tense of a word that we did not encounter in the

corpus or suppose you are finding capitalization in the middle of the set of the sentence you might think that this might be a proper name.

But how do we actually include that in hidden Markov models second. So, we use the first order Markov assumption where each tag dependent on the previous only on the previous tag. So, this does not take care of certain situations for example, here I have 2 sentences is clearly marked was such he clearly marked. So, what you seeing here marked in one case is a past participle in the first case second case it is a verb in past tense, but just the previous context is not sufficient to tell this. So, because in HMMs, we are only using the previous context this might not be sufficient to handle this particular case.

How will you take care of this issue? So, you might want to move to a higher order HMM say it second orders. So, you can use a high order model, but this is just one some limitations suppose you want to use what whether this is the first word in the sentence or you want to you want use any arbitrary think like whether, there is, a verb ending with e d previous to this word you want to use this information. There is no way you can use that in the case of achievements so that is where we move from achievements to my explanatory model that allows very very heterogeneous set of features. So, you can use a lot of different of your wishes from the data for your model.

(Refer Slide Time: 03:35)

*Maximum Entropy Modeling: Discriminative Model*

- We may identify a heterogeneous set of features which contribute in some way to the choice of POS tag of the current word.
  - Whether it is the first word in the article
  - Whether the next word is *to*
  - Whether one of the last 5 words is a preposition, etc.
- MaxEnt combines these features in a probabilistic model

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 1

What is the important distinction? So, now, in the maximum entropy model so they are discriminative as compared to generative models which are as means you are allowed to

identify very very heterogeneous set of features not only the tag given, the previous tag or the word given, the tag you can use, any sort of features which might which you think might contribute to the choice of part of speech tags.

For example, whether this is the first word of the article, this is something that you can use as a feature in maximum entropy model or whether the next word is ‘to’, you can use the next word as a feature whether one of the 5 previous 5 words is a preposition this is again something that you can use as a feature in maximum entropy model, but here HMM cannot make each of this.

What maximum model does it combines all this various heterogeneous features in probabilistic framework to be able to come up with the actual part of speech tag sequence for a given sentence.

(Refer Slide Time: 04:47)

*Maximum Entropy: The Model*

$$p_{\lambda}(y|x) = \frac{1}{Z_{\lambda}(x)} \exp\left(\sum_i \lambda_i f_i(x, y)\right)$$

where

- $Z_{\lambda}(x)$  is a normalizing constant given by

$$Z_{\lambda}(x) = \sum_y \exp\left(\sum_i \lambda_i f_i(x, y)\right)$$

- $\lambda_i$  is a weight given to a feature  $f_i$
- $x$  denotes an observed datum and  $y$  denotes a class

*What is the form of the features?*

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 3      4 / 16

This is the maximum entropy model.

$$P_{\lambda}(y|x) = (1/Z_{\lambda}(x))(\exp(\sum_i \lambda_i f_i(x, y))) \text{ where } Z_{\lambda}(x) = \sum_y \exp(\sum_i \lambda_i f_i(x, y))$$

So, this is how you write the probability of so  $y$  is the class that is one of the probability for any of the tags, given  $x$  is my observation it can include anything in my in my context it can load the current word, previous word, next word, anything in the sentence you can so you can define your context while you are dealing with maximum entropy model, you are saying that I can use all this information in my features, given this I want to compute the probability of

my class and here discriminative model. So, I can directly compute the probability of class given the observation. So, I can compute  $y$  given  $x$  and how do I compute that? It is simply  $x$  is a logging in model. So, summation over lambda  $i$  and  $f_i x y$ , now  $f_i x y$  are my various features that I have identified over my observation  $x$  and my class  $y$  and lambda is the weight given to the feature  $f_i$ . So, each feature  $f_i$  is given a weight lambda.

I summed over all the possible lambda is  $f_i$ 's and taking the exponent and then I do some sort of normalization. So, that probability  $y$  given  $x$  adds up to 1 for all the possible classes  $y$ , this is a normalization constant. So, this is the very very simple form of maximum entropy model. So, so let me tell again so  $z$   $x$  condition on the parameters lambda is normalization constant. So, this is nothing but this exponent you sum over all the possible class  $x y$  for all the possible class  $x y$  you compute this and add this. So, once you do that you are ensuring that probability  $y$  given  $x$  will added to 1 for all possible values and lambdas are my features the weights given to my different features.

Now  $x$  denotes my observed data and  $y$  and  $y$  denotes my class now, what is interesting is that? What is the form that the features in maximum entry model can take as opposed to something like HMM, what are the features? So, what is interesting here is you are seeing  $f$  is a function of  $x$  and  $y$  both in the observation in the class. So, I am defining my feature on both of this and we are all binary features.

(Refer Slide Time: 07:36)

*Features in Maximum Entropy Models*

- Features encode elements of the context  $x$  for predicting tag  $y$
- Context  $x$  is taken around the word  $w$ , for which a tag  $y$  is to be predicted
- Features are binary values functions, e.g.,

$$f(x,y) = \begin{cases} 1 & \text{if } \text{isCapitalized}(w) \& y = NNP \\ 0 & \text{otherwise} \end{cases}$$


Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 1

Let us see, what are my features? So, in maximum entropy model, my feature send code elements from the context  $x$  and the tag  $y$  now context  $x$  is now this is something that you must be careful about context is not my current word  $w$ , but is taken around the word  $w$  i can define my context it can be the sentence that contains  $w$  it can be it is previous 3 words next 3 words it is something that can be defined on the on the current word and  $y$  is the tag that I have, I am predicting

Now, what are my features these are binary valued functions over  $x$  and  $y$  and this is like  $f(x, y)$  is 1 if a certain conditions condition hold on  $x$  and  $y$  and 0 otherwise and here is one example what can be some conditions like I am having the current word  $w$  if the word  $w$  is capitalized and my tag  $y$  is NNP then my feature  $f(x, y)$  will take a value of 1 otherwise it will take a value of 0 that is one sort of feature. Similarly you can define any arbitrary set of features that we have said, it can be  $f(x, y)$  is equal to 1 if one of the previous 5 words is as a tag of preposition and the current tag is save up this is one sort of a 0 otherwise. So, like that you can define an. So, you have a lot of freedom here in defining and choosing your features and once you complete all these features the model learns the weights for all these feature features the weights lambdas and then this is very very simple function to compute the probability of plus given any observation.

(Refer Slide Time: 09:18)

- Example: Named Entities
- LOCATION (in Arcadia)
- LOCATION (in Québec)
- DRUG (taking Zantac)
- PERSON (saw Sue)

 Below this, there is another section titled 'Example Features' with three mathematical definitions of features:
 

- $f_1(x, y) = [y = \text{LOCATION} \wedge w_{-1} = "in" \wedge \text{isCapitalized}(w)]$
- $f_2(x, y) = [y = \text{LOCATION} \wedge \text{hasAccentedLatinChar}(w)]$
- $f_3(x, y) = [y = \text{DRUG} \wedge \text{ends}(w, "c")]$

 The slide footer includes the name 'Pawan Goyal (IIT Kharagpur)', the topic 'POS Tagging', and the week 'Week 4, Lecture 1'. A small circular video player shows a person speaking."/>

Let us see some other examples for the features. So, this is again for any generic sequence labeling task. So, here it is for finding named entities. So, it can be here like location drug and

person 3 different named entities and some examples are given. Like in arcadia, arcadia is a location in Quebec location taking Zantac, Zantac is drug and saw sauce you. So, you see you as a person and now you are defining certain features. So, let us see certain features like - the first feature is my tag is location. So, my class y is location previous word that is on x previous word is in and my current word is capitalized.

What are all the cases we have? This feature will be one, it will be one here in arcadia previous what is in and it is capitalized and the tag is location, it will also be one for second case is capitalized previous word is in and the tag is location. Next feature, y is location and w has an extended leg in Latin character. So, you can see that you are using a very very arbitrary set of features, but possibly you have some hypothesis that this features will be helpful for this task operating the predicting the tag for this word. So, this is this feature. So, for which of the case it will be? One, it will be one for the second case it has an extended Latin character and the tag is location.

Third feature, y is a drug and the word ends with c and there will be one for taking Zantac yes the word W ends to the c. So, you can see some examples of what kind of features you can take, but important thing is to know the intuition that you can choose any feature around the word w and that involves also the current class.

(Refer Slide Time: 11:26)

### Tagging with Maximum Entropy Model

- $W = w_1 \dots w_n$  - words in the corpus (observed)
- $T = t_1 \dots t_n$  - the corresponding tags (unknown)

Tag sequence candidate  $\{t_1, \dots, t_n\}$  has conditional probability:

$$P(t_1, \dots, t_n | w_1, \dots, w_n) = \prod_{i=1}^n p(t_i | x_i)$$

- The context  $x_i$  also includes previously assigned tags for a fixed history.
- Beam search is used to find the most probable sequence

Navigation icons

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 3      7 / 16

$$P(t_1, \dots, t_n | w_1, \dots, w_n) = \prod_{i=1}^n p(t_i | x_i)$$

How do we? So, now, once I have all these features suppose the learning has taken place I know all the parameters lambdas, how do I go about tagging my sentence with that  $x$ ? So, I have the words in my corpus  $w_1$  to  $w_n$  and I want to find out the tags document tags. So, what I will do? I want to find out tags sequence  $t_1$  to  $t_n$ . Any sequence will have this condition, probability right. So  $t_i$  conditioned on  $x_i$  for  $i$  is equal to 1 to  $I$ , now this is  $x_i$  not  $w_i$ ,  $x_i$  is my context around the word  $w_i$  and this is independent.

So, I can use this model to find out what is the optimal tag sequence, I can do it independently for each word and then I will find out tag sequence. So as we have said my context  $x_i$  includes, may also include some previously assigned tags for a fixed history and I will be using a beam search algorithm for finding the optimal or the most probable sequence this is something we will take up in the next lecture in detail that how do we actually come go about doing beam search in maximum entropy model.

(Refer Slide Time: 12:51)

Beam Inference

Beam Inference

- At each position, keep the top  $k$  complete sequences
- Extend each sequence in each local way
- The extensions compete for the  $k$  slots at the next position

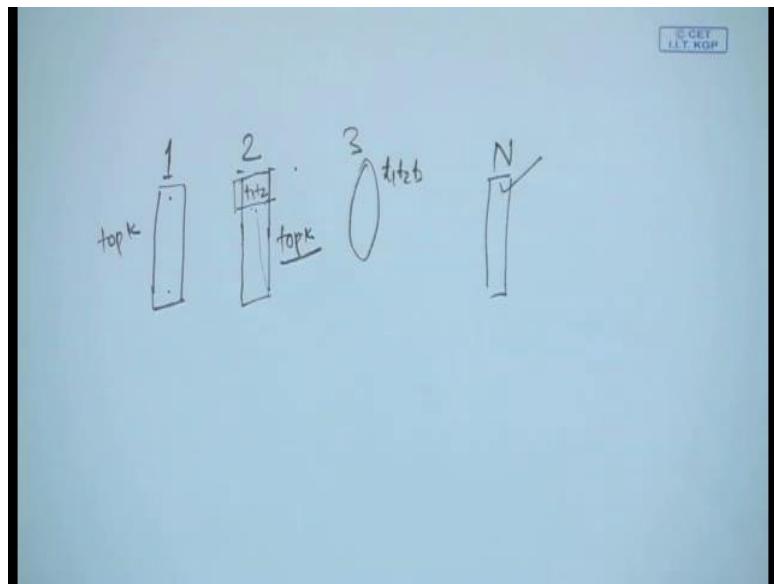
But what is a MaxEnt model?

Let's go to the basics now!

Pawan Goyal (IIT Kharagpur) POS Tagging Week 4, Lecture 3 8 / 16

Yeah just a brief idea, here in beam search what you do actually. So, you have positions in the sentence a starting from one to the end. So, at each position we will keep only the top  $k$  possible sequences and next time starting from this  $k$ ;  $k$  complete sequences you find out what are the next  $k$  based sequences.

(Refer Slide Time: 13:28)



Let me just quickly say, so I have from, suppose my sentence as  $N$  words so, what is the idea? Suppose my size beam size is  $k$ , at point 1 I will keep top  $k$  sequences. So, at point 1, I will have only 1 part of speech tags so; that means, I will keep top  $k$  part of speech tags at it is position 1, now when I go to position 2, now I will have a sequence, now 2 might have some other possibilities of part of speech tags. So, what I will do? I will find out the probability of each of this  $k$  with all part of speech tags here and you store top  $k$  sequences again. So, tag 1, tag 2 some case top  $k$  sequences again at 3, I have some possibilities top  $k$  times these probabilities this possibilities again you store only top  $k$  out of this and you keep on doing that until you are here.

Here you will choose the best one and that will give you the complete sequence. So, important here is that at each point you are storing top  $k$  full sequences which starting from the initial point. So, here you will find some  $t_1, t_2$ , you will find some  $t_1, t_2, t_3$  and so on top  $k$  at it is time. So, we will look at this in detail in the next lecture. So, for now let us look at the basics of maximum entropy model. So, what is a maximum entropy model?

(Refer Slide Time: 15:02)

**Maximum Entropy Model**

**Intuitive Principle**

Model all that is known and assume nothing about that which is unknown.  
Given a collection of facts, choose a model which is consistent with all the facts, but otherwise as uniform as possible.

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 3      9 / 16

Maximum entropy model have this particular principle that is you are given the observation and you have certain hypothesis. So, you take what is given to you, but do not make any additional assumptions. So, that is take the model that satisfies all the questions that you having, but does not take any other assumptions.

What does that mean? So, you are given a collection of facts yes and you want to take a model that is satisfying all these facts or you are saying that is consistent with all these facts, but otherwise it is not making any further assumption that is after being after satisfying all these constraints it is as uniform as possible or in other words for all the possible models that satisfy a given set of constraints choose the one that is the most uniform of all.

(Refer Slide Time: 16:09)

**Maximum Entropy: Overview**

- Suppose we wish to model an expert translator's decisions concerning the proper French rendering of the English word 'in'.
- Each French word or phrase  $f$  is assigned an estimate  $p(f)$ , probability that the expert would choose  $f$  as a translation of 'in'.
- Collect a large sample of instances of the expert's decisions
- **Goal:** extract a set of facts about the decision-making process (first task) that will aid in constructing a model of this process (second task)

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 3      10 / 16

Now, what do I mean by this? Let us take a simple example. So, here I have an, this I have a simple example I have an English word in and I want to make some transition system that tries to mimic some. So, it is translating from English to French and I have some data already with me from some actual translator who translated some sentences from English to French.

Now, so I want to model this expert translator's decision that how do I translate this word in to any French word? Now how do I start? So, suppose you have a vocabulary of all possible French phrases and you want to estimate probability  $p(f)$ . So, that is, what is the probability that that French phrase will be a translation of the English word in; so you want to compute that for all the possible French phrases, they might be say, millions of French phrases, so what you will do? We will start by collecting a large sample of instances of expert decisions.

Now, from this facts you will try from this sample you will try to observe some facts for example, so the goal would be extract some facts from the sample of instances and then use these facts inside your model so. Firstly, you have to extract the facts. So, in our scenario this will be equivalent to finding out the features that is my facts and once I found these features or the facts I use them rightly in my model.

(Refer Slide Time: 18:06)

*First clue: list of allowed translations*

- Suppose the translator always chooses among {dans, en, á, au cours de, pendant}.
- First constraint:  $p(\text{dans})+p(\text{en})+p(\text{á})+p(\text{au cours de})+p(\text{pendant}) = 1$ .
- Infinite number of models  $p$  for which this identity holds, the most intuitive model?
- allocate the total probability evenly among the five possible phrases → most uniform model subject to our knowledge.
- Is it the most uniform model overall?

Pawan Goyal (IIT Kharagpur) POS Tagging Week 4, Lecture 1

Let us see what are the different sort of facts I can find out for example, this can be my first clue. So, I find that in all the samples I have seen the translator always translates in into one of these 5 phrases, dans, en, a, au cours de, pendant. So, they are 5 different French phrases where the translator is translating. So, now, this is helpful to explain what is my maximum entropy model.

(Refer Slide Time: 18:40)

1 million, f<sub>m</sub> P(f)

① 
$$P(f_1) + P(f_2) + P(f_3) + P(f_4) + P(f_5) = 1$$
 give

$P(f_1) \dots P(f_5) = 0$  Most Uniform

$\therefore P(f_i) = \frac{1}{5} \quad i=1-5$

② 
$$P(f_1) + P(f_2) = 0.3$$
  
 $P(f_1) = 0.15 \quad P(f_2) = 0.15$

Now, I want to find out the probability of the probability distribution over French phrases all. So, suppose I have say 1 million French phrases, so, f 1 to f million. So, I have 1 million

French phrases, I want to compute this probability distribution, I want to find out the probability distribution. Given this constraint that is the translator always chooses among from one of these 5 phases. So, what is my constraint? This would be so let me call these  $f_1$ ,  $f_2$ ,  $f_3$ ,  $f_4$  or say  $dans$  and also let me call these probability say  $f_1$  plus probability  $f_2$  plus probability  $f_3$  plus probability  $f_4$  plus probability  $f_5$  is equal to so, this is my constraint.

Now, given this constraint I have to choose 1 model. So, suppose I have only this much information from the data now what is the model that I should choose among all the possibilities now? Firstly, do you think there are infinite number of possibilities in which I can choose my model that satisfy this constraint. So, of course, probability for  $f_6$  to probability  $f_{\text{million}}$  will be 0 because if any if any of this is non 0 this constraint will not be satisfied.

Yes, so, because all the probabilities need to add up to one so; that means, all this has is this is anyways easy, but what are all the models that will satisfy this constraint. So, you can see there are infinite solutions for this equation. So, you can choose  $p(f_1)$  any number from 0 to 1 and you can always find some  $f_2, f_3, f_4, f_5$  that will satisfy and same goes for  $f_2$  and so on. So, there are infinite models that satisfy this constraint.

Now, what maximum entropy model does among all the models that satisfy this constraint it takes the one that is most uniform, now among all these models what is the most uniform model. So, that is you are not making any dimensions given this constraint take the model that is most uniform. So, it is easy to see in this case that the most uniform model will be one that gives  $P(f_i)$  is equal to 1 by 5 for  $i$  is equal to 1 to 5, for all these 5 phrases gives the probability 1 by 5 and everything else it gives a probability 0. So, this is the most uniform model given this constraint. So, allocate the total probability evenly among all these 5 phrases that is the most uniform model subject to our knowledge.

Now is this the most uniform model overall, suppose I did not have this constraint what was the most uniform model as such, the most uniform model will be that gives equal probability to each of the million French phrases. So, it is gives the probability of 1 by million to each of these that is a most uniform model. So, the model that I have obtained which is giving probability 1 by 5 to each of these 5, it is not the most uniform model, it is most uniform given this constraint given this constraint this is the most uniform model and this is the idea.

(Refer Slide Time: 22:31)

*Maximum Entropy Model: Overview*

*More clues from the expert's decision*

- **Second clue:** Suppose the expert chose either 'dans' or 'en' 30% of the time.
- **Third clue:** In half of the cases, the expert chose either 'dans' or 'á'.

*How do we measure uniformity of a model?*

As we add complexity to the model, we face two difficulties:

- What exactly is meant by "uniform"?
- How can one measure the uniformity of a model?

Ptwan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 1

I will take some facts some observations given the observations I will find out what is the most uniform model.

Let us take; let us go further. So, this was easy in this case. So, where we had 5 phases only knows i got a second clue that the expert chose either dans or en 30 percent of the time. So, we were supposed f 1 and f 2. So, now, I have, this is my constraint one. So, now, i have constraint 2 P f 1 plus P f 2 is equal to 0.3. Now given these 2 constraints, what would be the most uniform model? So, if you just take a quick look you can see that the most uniform model will be P f 1 is equal to 0.15 P f 2 is equal to 0.15 and f 3 f of 5 f 5 which will get 0.7 divided by 3 and that will set both the constraint this will be the most uniform model. So, this is not as uniform as this model. So, as you are getting more and more observations, you are going far from the most uniform models, but you are coming up with some other models that I as uniform as possible, but satisfying these constraints.

Now, what happens if you get a third clue like in half of the case; cases expert shows either dans or an. So, that is P f 2 plus P f 3 is equal to 0.5 that is 3, you know immediately you will see that it becomes difficult to just look at this equation and try to solve for what is the distribution that is most uniform and that is why we need to understand the concept of maximum entropy. So, I want to find out the model that is most uniform doing the constraints that is equivalent to finding the model that is having the maximum entropy among model.

Now, yes how do we? So, as we get more and more clues it becomes difficult to see which is the most uniform model. So, for that you have to understand how do we measure the uniformity of a model and that is where we use the idea of entropy.

(Refer Slide Time: 24:49)

**Maximum Entropy Modeling**

**Entropy:** measures the uncertainty of a distribution.

*Quantifying uncertainty ("surprise")*

- Event  $x$
- Probability  $p_x$
- Surprise:  $\log(1/p_x)$

*Entropy: expected surprise (over  $p$ )*

$$H(p) = E_p \left[ \log_2 \frac{1}{p_x} \right] = - \sum_x p_x \log_2 p_x$$

Coin Tossing

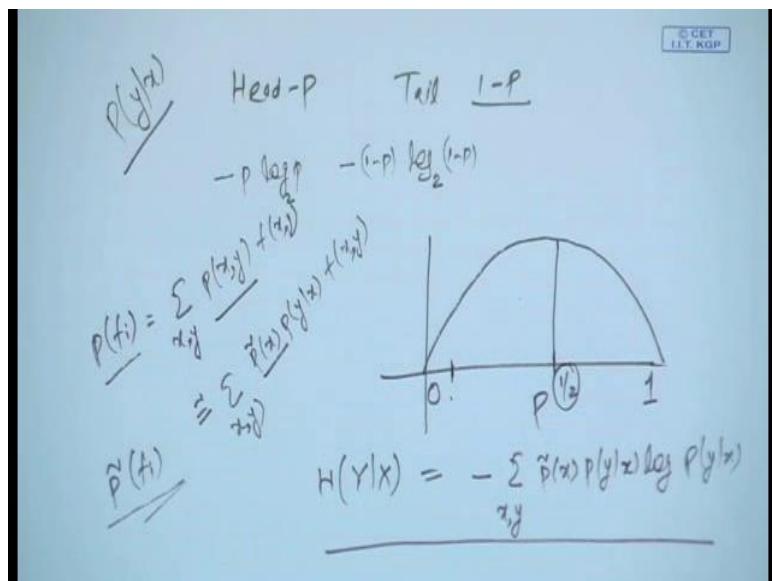
Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lect 1

So, some of you might have already come across this term of entropy. So, what are entropy measures? What is the uncertainty of distribution? So, going to the basics, so suppose I have an event  $x$  that occurs to the probability of  $P_x$ .

Entropy measures what is the amount of surprise in this event. So, if it is very very surprising then the entropy will be high, if it is like very obvious then entropy will be low so, for an event when it is more surprising when the probability is low, probability is high and the probability hi high it is like obvious. So, then the entropy will be low, but when the probability when the; so the interview will be low, but when the probability is low. So, there is a lot of surprise in that even if that happens then the probability entropy is high. So, that is why entropy is log of 1 by probability.

The probability is low entropy will be high. So, for a distribution how do we compute the entropy for a distribution? So, I just take the expected value of surprise that is summation over  $p_x \log 1 by P_x$ . So, I take the expected surprise over all the possible values of  $P$ . So, this is a very very a standard formula for entropy minus summation  $P_x \log P_x$  now. So, here we are talking about distributions in the case of distributions when will be the entropy maximum. So, if you look at this formula.

(Refer Slide Time: 26:58)



Let us take 1 simple example of coin tossing. So, I am tossing a coin. So, it can be head or tail let us say that head has a probability of  $P$  and tail has a probability of  $1$  minus  $P$ . So, what is the entropy of this model? It will be  $-\sum p(x,y) \log p(x,y)$ . If you actually go on plotting this as a function of  $P$  from  $0$  to  $1$  you will find that the entropy it starts increasing it becomes the highest at  $P$  is equal to half and it starts decreasing afterwards, now can we intuitively understand this?

What we are saying when both head and tail have equal probability then the entropy of them of my model is the highest, what is entropy? It measures how much is the surprise. So, let us take a point like this. So, at this point what will happen? So, head has a probability very low probability, but tail as a very high probability. So, you more or less no it will probably tail when you toss a coin it is probable most probably be a tail. So, there is not, there is not much more surprises that are coming, but when both head and tail have equal probability then you are very confused you do not know whether the it will be head or it will be tail. So, the amount of so, expected surprise that you will get is the maximum. So, this is the idea. So, this is a distribution over only this 2 variables  $P$  and this  $P \times Q$  are you can say this have  $P \times Q$  minus  $P$ .

But even if you take any  $n$  number of variables the reservation will be having the maximum entropy wave whenever it is the most uniform. So, this gives such the intuition that why we chose the most uniform model for getting the maximum entropy.

(Refer Slide Time: 29:06)

**Maximum Entropy Modeling**

**Distribution required**

- Minimize commitment = maximize entropy
- Resemble some reference distribution

**Solution**

Maximize entropy  $H$ , subject to feature-based constraints:

$$E_p[f_i] = E_{\bar{p}}[f_i]$$

**Adding constraints**

- Lowers maximum entropy
- Brings the distribution further from uniform and closer to data

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 1

A small circular video player shows a person speaking.

Now, coming back to a maximum entropy model, so we want to, among all the possible distributions we want to choose a model that maximizes by entropy.

Now, so I want to maximize entropy subject to certain constraints. So, we will have certain constraints from the data and these constraints are nothing, but my features. So, I will select some features and I will make sure that the ambiguous probability from the data that I am finding for the features is similar to the probability that my model is giving that and subject to this constraint I will maximize my problem my entropy. As you also seen as we keep on adding constraints one by one, the entropy of my model decreases, it comes close to my data initially might start with the most uniform model that is giving the equal probability to everything, but as you keep on adding constraint one by one you come close you go far away from the most uniform model. So, you decrease your entropy and you come closer to data. So, this is the idea. So, give us some constraints we want to come as close as possible to the data, but otherwise we want to obtain the most uniform model in terms of maximum entropy.

(Refer Slide Time: 30:36)

**Maximum Entropy Principle**

Given  $n$  feature functions  $f_i$ , we would like  $p$  to lie in the subset  $C$  of  $P$  defined by

$$C = \{p \in P \mid p(f_i) = \tilde{p}(f_i), i \in \{1, 2, \dots, n\}\}$$

*Empirical count (expectation) of a feature*

$$\tilde{p}(f_i) = \sum_{x,y} \tilde{p}(x,y) f_i(x,y)$$

*Model expectation of a feature*

$$p(f_i) = \sum_{x,y} p(x)p(y|x)f_i(x,y)$$

Select the distribution which is most uniform (conditional probability):

$$p^* = \operatorname{argmax}_{p \in C} H(p) = H(Y|X) \approx - \sum_{x,y} \tilde{p}(x)p(y|x)\log p(y|x)$$

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 3      15 / 16

Coming close to the data in this case means finding the empirical probabilities of these features and I am trying to make it trying the model probabilities to be close to this. So, this is my maximum entropy principle

I am given some  $n$  different

$$C = \{p \in P \mid p(f_i) = \underline{p}(x,y) f_i(x,y)\}$$

feature functions  $f_1$  to  $f_n$ , I want my probably distribution  $p$  to lie in the subset in the subset of all the possible distributions that satisfy this constraint. So, all the probability distributions that satisfy this constraint among those I want to choose the one that is having the maximum entropy. So, what do I mean by satisfying this constraint. So, what is the empirical count of or expectation of a feature? So, empirical expectation means how many times this features actually this  $x$   $y$  actually occurred and  $f_i(x,y)$  was 1 in my data. So,  $\hat{P}$  is my empirical probability.

$$\underline{p}(f_i) = \sum_{x,y} \underline{p}(x,y) f_i(x,y)$$

Now, what will be my model probability? So, it will be so model probability  $P(f_i)$ , it will be summation  $x$   $y$   $P(x,y)$ ,  $f_i(x,y)$  now this  $P(x,y)$  because we are computing finally,  $P(y)$  given  $x$  I can write as summation  $x$   $y$   $p(x)p(y)$  given  $x$   $f_i(x,y)$  and because we are not computing  $P(s|n)$  model I can probably approximate it with the empirical count of  $x$ . So, this gives me the model probability of the feature. So, I want to make it closer to the  $\tilde{p}(f_i)$  that is the empirical probability of the feature.

I have the empirical count model count and I want to select the distribution that is most uniform subject to this constraint. So, I am computing this probability distribution. So, what will be the maximum entropy model that is the one that gives me the high entropy  $H(y)$  given  $x$  and that if you do some quickness that will be minus  $\sigma_{x,y} P_x P_y$  given  $x \log P_y$  given  $x$  and again we will approximate this by  $P_{\tilde{x}}$ . So, this is my maximum entropy model.

(Refer Slide Time: 33:25)

### Maximum Entropy Principle

$$p^* = \operatorname{argmax}_{p \in C} H(p)$$

#### Constraint Optimization

Introduce a parameter  $\lambda_i$  for each feature  $f_i$ . Lagrangian is given by

$$\Lambda(p, \lambda) = H(p) + \sum_i \lambda_i (p(f_i) - \bar{p}(f_i))$$

Solving, we get

$$p_i(y|x) = \frac{1}{Z_i(x)} \exp\left(\sum_i \lambda_i f_i(x, y)\right)$$

where  $Z_i(x)$  is a normalizing constant given by

$$Z_i(x) = \sum_y \exp\left(\sum_i \lambda_i f_i(x, y)\right)$$

Ptwan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 3      16 / 16

And I want to maximize it subject to constraint that this and these are equal. So, this gives me the models maximize this  $H$  subject to these constraints.

(Refer Slide Time: 33:48)

The image shows a handwritten derivation of the maximum entropy model. It starts with the constraint  $\sum_y p(y|x) = 1$  and the expression for the lagrangian  $\lambda$ :

$$\frac{\partial \lambda}{\partial t} = 0$$

$$\lambda = -\sum_{yj} \tilde{p}(y|x) \log \frac{p(y|x)}{t}$$

$$\lambda(p, \lambda) = H(p) + \sum_i \lambda_i (p(t_i) - \tilde{p}(t_i))$$

The derivation involves differentiating the lagrangian with respect to  $t$  and setting it to zero, leading to the final expression for the lagrangian.

For that we use the trick of being the lagrangian. So, I want to now a k lagrangian P or lambda that is my H P plus summation i lambda i P f i minus P tilde f i. So, that is I want to maximize this sorry. So, I want to maximize this yes subject to this constraint.

Now if I put these values inside this so I will get - plus sigma lambda i summation x y P x y stepwise P tilde x P y given x f x y minus summation x y P tilde x y f x y, this is my lagrangian and I want to maximize that. So, I want to find out this distribution that maximizes this. So, let me just substitute this (Refer Time: 35:58) t everywhere, so in turn what you are doing? You are maximizing it with respect to this t and that you can do by differentiating it with respect to t and then putting it into 0 and that will actually give you the equation that we have seen; that is p lambda y given x that is t over in this place it will come out to be exponential over summation lambda f i x y divided by normalization constant.

This is simple exercise that you can do you can just differentiate it with respect to this parameter t and you will actually come up with something close to what we have seen in the formulation of maximum entropy model. And this normalization constant is nothing but something that makes this probability p lambda y given x summation over y to 1 and that gives me the normalization constant. This is my maximum entropy model.

So, you have to, the important part here is to find out the set of features once you have a set of features you can use this particular equation to compute the probability of a class y given the observation x.

Next, in the next lecture, I will try to take up is a problem that will make the idea of maximum entropy model for that clear then we will see how given a new sentence we use this maximum entropy model to come up with the tag sequence and there we will study this beam algorithm, beam search algorithm. So, see you then.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 20**  
**Maximum Entropy Models – II**

So, welcome back for the fourth module of this week. So, in the last module, we had started talking about the maximum entropy classifiers what is the basic principles for using this maximum entropy classifiers. And we saw that we can use any hidden new set of features that were not allowed in the earlier model like HMM. So, today I will start by taking a simple example that will give you some idea on how maximum entropy classifier actually works for a simple classification problem. And then we will see how it can be used for a sequence tagging problem that is I am given a sentence. So, this is my problem for part of speech tagging and given the sentence for each of the words, I have to find out what is the actual part of speech tag for that.

So, right now what we had seen in maximum entropy classifier, it gives me a probability on the class  $y$  given a context. So, for each individual word for the context around that word, I can predict what is the appropriate part of speech category, but once I am given the sequence how this model will be actually utilized. So, there it has a special name called MEMM - maximum entropy Markov model and we will see what is the algorithm for using the classifier that we have discussed for a sequence tagging problem.

(Refer Slide Time: 01:40)

**Practice Question**

Consider the maximum entropy model for POS tagging, where you want to estimate  $P(\text{tag}|\text{word})$ . In a hypothetical setting, assume that *tag* can take the values *D*, *N* and *V* (short forms for Determiner, Noun and Verb). The variable *word* could be any member of a set *V* of possible words, where *V* contains the words *a*, *man*, *sleeps*, as well as additional words. The distribution should give the following probabilities

- $P(D|a) = 0.9$
- $P(N|\text{man}) = 0.9$
- $P(V|\text{sleeps}) = 0.9$
- $P(D|\text{word}) = 0.6$  for any word other than *a*, *man* or *sleeps*
- $P(N|\text{word}) = 0.3$  for any word other than *a*, *man* or *sleeps*
- $P(V|\text{word}) = 0.1$  for any word other than *a*, *man* or *sleeps*

It is assumed that all other probabilities, not defined above could take any values such that  $\sum_{\text{tag}} P(\text{tag}|\text{word}) = 1$  is satisfied for any word in *V*.

- Define the features of your maximum entropy model that can model this distribution. Mark your features as  $f_1, f_2$  and so on. Each feature should have the same format as explained in the class.  
[Hint: 6 Features should make the analysis easier]
- For each feature  $f_i$ , assume a weight  $\lambda_i$ . Now, write expression for the following probabilities in terms of your model parameters
  - $P(D|\text{cat})$
  - $P(N|\text{laughs})$
  - $P(D|\text{man})$
- What value do the parameters in your model take to give the distribution as described above. (i.e.  $P(D|a) = 0.9$  and so on. You may leave the final answer in terms of equations)

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 4      2 / 8

So, now starting with a simple problem on how to use maximum entropy classifier, the practice problem, so you can try it on your own but I will try to give you certain hints. So, problem says, so you want to use maximum entropy model for part of speech tagging, and you are trying to estimate probability tag given word. And in this hypothetical setting, you can assume that there are only three possible tags determiner, noun and verb. And the word can belong to any member of a set *V* that is your vocabulary; and vocabulary contains the words *a*, *man*, *sleeps* and some additional words. So, in this problem it does not matter how many words are there in your vocabulary.

Now, the constraint given is as follows. So, the distribution that you will find probability tag on word should give the following probabilities. Probability of determiner given the word *a* is 0.9, probability of noun given the word *man* is 0.9, tag *V* given *sleeps* is 0.9, *D* given any word other than *a*, *man* or *sleeps* in the vocabulary is 0.6 and same for that two constraints that are given here. So, remember that is what we did in the case of maximum entropy classifier. You are given the data; and from the empirical distribution, you extract certain facts and now you want to find out a distribution that resembles that empirical distribution.

So, now I want to find out probability *I* given word such that these constraints are followed. So, now how do I go about building my maximum entropy classifier this is the problem. So, now it is said that all other probability that are not given suppose that take some values such

that probability I given word is one for all possible tags, this is the normalization constraint, the probability for all the possible tags for a given words should add up to 1.

Now, there are certain questions here. So, the first question says is that so you are given this problem settings. Now, try to define what will be the features option maximum entropy model given these constraints. And the problem says that Markov features as some  $f_1$  and  $f_2$ . And each feature should have the same format as we explained in the last class, so that is each feature your function of  $x$  and  $y$   $x$  is the context around the word and why is the particular class or tag.

Then the next question is for each feature  $f_i$  assume a weight of lambda I now write down the expression of the following probabilities. So, what is the probability of D given cat in terms of your model parameters? So, the parameters are mainly lambda 1 to lambda 6. Similarly, find out probability of N given laughs probability of D given man. Finally, the problem is what should be the values of the parameters in your model lambda 1 to lambda 6 such that you have to the distribution as shown in the question. So, this is by problem. So, let me try to solve it n-steps. So, this is the first problem what should be the features of my model, such that they resemble this distribution. So, these are the actually certain facts from the data.

(Refer Slide Time: 05:01)

The image shows handwritten notes on a whiteboard. At the top, it says "P(D/a) = 0.9". Below this, it defines a function  $f(x,y)$  where  $x$  is the word context and  $y$  is the tag. It shows two examples:  $f_1: f_1 = \text{word} = 'a' \& \text{tag} = 'D' - f_1(\lambda)$  and  $f_2: f_2 = \text{word} = 'man' \& \text{tag} = 'N'$ . Further down, it shows  $f_3: f_3 = \text{word} = 'X' \& \text{tag} = 'V'$  and  $f_4: f_4 = \text{word} \in V \& \text{tag} = 'D'$ . There is also a note "- f\_5: f\_5 = \text{word} \in V \& \text{tag} = 'N'" and "- f\_6: f\_6 = \text{word} \in V \& \text{tag} = 'V'". To the right, it shows  $P(N/man) = 0.9$ .

So, let me write down the first fact probability D given a is 0.9. So, now I want to find out some features  $f_x, y$  such that it can try to match this particular constraint. So,  $x$  is on my data.

So, this was my word and the context and  $y$  is the tag. So, now what am I observing in this particular example I am observing that when the word is  $a$ , then the tag that I gives is  $D$  with the probability of 0.9. So, what kind of feature will try to model this? So, my feature should be affects  $y$ . So, I can use a feature word that is my  $x$  is equal to  $a$ , and tag that is my  $y$  is equal to  $D$  that encodes both  $a$  and  $D$  that are that I am observing in this empirical data. In this, we can call as your  $f_1$  what is  $a$  and tag is  $D$ . So, this is a binary feature. So, whenever you will see word as  $a$  and tagged  $D$ , you will put it as 1. So, to express it you can say  $f_1$  is equal to 1; if this happens is equal to 0; if this does not happen I am not writing it in this paper.

So, now I have to define some other features right and the hint is that six feature will make my task easy, so that is like sufficient hint you have six possible constraints and you have to define six feature one for each for each constraint. So, let us take the second constraint. Second constraint is probability  $N$  given man is 0.9. So, for this constraint, what will my feature, again so  $f_2$  would be word is equal to man and tag is equal to  $N$  – noun. Same way you can do for your third case. So,  $f_3$  will come out to be word is equal to sleeps and tag is equal to  $V$ .

Now, what would you do for your fourth observation? What is that probability  $D$  given word is equal to 0.6, for word in vocabulary minus  $a$ ,  $man$  and  $sleeps$ , for any word other than  $a$  minus  $sleeps$ . So, what would your feature, again you will have a similar feature  $f_4$  word is in, so let me call this as  $V'$   $V$  minus  $a$   $man$   $sleep$ , let me call it as  $V'$  this is my  $V'$  prime. So, word is in  $V'$  and tag is equal to  $D$ . Similarly, my  $f_5$  will become word in  $V'$  prime and tag is  $N$ .  $f_6$  word in  $V'$  prime and tag is  $V$ , and these become my six features that will try to model the distribution. So, this is how you will select the features given certain facts about the data.

So, see here I am not doing anything from matching this, these numbers right now, so that I will try to do using my lambdas that the feature weights set that I have to choose. So, the next part of the question says is that now give weights. So, suppose this is lambda 1, it has a weight of lambda 1, lambda 2, lambda 3, and so on. So, each feature  $f_i$  has a weight of lambda  $i$ . Now, try to find out some probabilities in terms of the model parameters. So, let me take out the first question itself that is probability of  $D$  given  $cat$ , I want to write that in terms of model parameters.

(Refer Slide Time: 10:04)

The image shows a handwritten derivation of the probability  $P(D|\text{cat})$  using a maximum entropy model. The derivation starts with the formula:

$$P(D|\text{cat}) = \frac{e^{\sum \lambda_i f_i}}{Z}$$

Below this, the numerator  $\sum \lambda_i f_i(x,y)$  is expanded as:

$$\sum \lambda_i f_i(x,y) = \lambda_1 \cdot 0 + \lambda_2 \cdot 0 + \lambda_3 \cdot 0 + \lambda_4 \cdot 1 + \lambda_5 \cdot 0 + \lambda_6 \cdot 0$$

The denominator  $Z$  is shown as:

$$Z = \frac{e^{\lambda_5}}{Z} + \frac{e^{\lambda_6}}{Z} = \frac{e^{\lambda_4}}{Z} = \frac{e^{\lambda_4}}{e^{\lambda_4} + e^{\lambda_5} + e^{\lambda_6}}$$

Finally, the probability is given as:

$$P(N|\text{cat}), P(V|\text{cat}) \left[ \frac{e^{\lambda_4} + e^{\lambda_5} + e^{\lambda_6}}{Z} = 1 \right]$$

So, what will be this probability D given cat? So, now in terms of my maximum entropy model, how do I write this particular probability, it should be  $e$  to the power sigma lambda if i divide by a normalization constant; and Z should be such that this will add up to one for all the three tags. So, probability D given cat plus probability N given cat plus probability V given cat should be 1. So, let me take this particular y, let me focus only on this and I will talk about Z later. So, now so this is sigma lambda i f i x y, x here is cat and y is D. So, now f x, y, so each of the six features are binary features, so they will take either 1 or 0; and lambda ones are just given to me and then the values are not given. So, I can just take lambda 1, lambda 2 and so on.

So, now, what will be this value? So, let me take the first feature. So, first feature is here word is a, and tag is D. In this case, my word is cat, so clearly this feature is not one. So, for f 1, x 5 it will be 0. So, lambda 1 times 0 plus lambda 2 times; what is my second feature second feature says that word is man and tag is n again this is 0; f 3 word is sleeps tag is V again 0. F 4 so my feature f 4 is word is in v prime, is this word in v prime cat is not either a man and sleep so it is in v prime, and tag is d. So, this is actually one. So, word is in v prime and tag is d, so lambda both 4 times 1 is I will not write f 4, f 4 is 1 in this case plus lambda 5. Now, you will see in feature 5 and feature 6, the tags are different they are N and V. So, they will again be 0 plus lambda 6 0, this comes out to be lambda 4. So, now what is my probability? It will be  $e$  to the power lambda 4 divided by Z.

Now how do you find out  $Z$ ? Now have to find out  $Z$ , you have to actually compute the other two probabilities also. So, you have to compute probability  $N$  given  $\text{cat}$ , probability  $V$  given  $\text{cat}$  and then you can make this plus this plus this is equal to 1, so let me quickly do that. So, what will happen for when you compute probability  $N$  given  $\text{cat}$ ? The first three features again use a word  $a$ ,  $\text{man}$  and  $\text{sleeps}$  that is not here. From fourth features, they are using the words in  $V$  prime, so the word  $\text{cat}$  is in  $V$  prime. So, the features can be 1 from  $f_4, f_5, f_6$ , but now I have to see the tag. For  $f_4$ , the tag was  $D$ ; for  $f_5$ , the tag was  $N$ ; and for  $f_6$ , the tag was  $V$ ; this was  $N$  and this was  $V$ . So, for  $N$  given  $\text{cat}$   $f_5$  will be 1; and  $V$  given  $\text{cat}$   $f_6$  will be 1. So, can I write probability  $N$  given  $\text{cat}$ , this will be  $e$  to the power  $\lambda_5$  given  $Z$ , and this will be  $e$  to the power  $\lambda_6$  divided by  $Z$ .

Now, you can compute what is  $P(D \text{ given } \text{cat})$  because this  $c$  will add up to 1. So, put in the constraint  $e$  to the power  $\lambda_4$  plus  $e$  to the power  $\lambda_5$  plus  $e$  to the power  $\lambda_6$  divided by  $Z$  is equal to 1, so that gives you  $Z$  is equal to  $e$  to the power  $\lambda_4$  plus  $e$  to the power  $\lambda_5$  plus  $e$  to the power  $\lambda_6$ . So, this we can write as  $e$  to the power  $\lambda_4$  divided by  $\lambda_4 + \lambda_5 + \lambda_6$  that is your probability of  $D$  given  $\text{cat}$  in terms of your model parameters.

Now, in a very similar manner, I will suggest that you try out the other two cases, what is the probability of  $N$  given  $\text{laughs}$ , and what is the probability of  $D$  given  $\text{laugh}$ . So, again we have not use of the probabilities 0.9 and etcetera that were given in the constraint of discussion, so that I have to use in the last point. So, what values with the parameters in your model take to give the distribution as described above, so that is what values they should take such that probability  $D$  given  $a$  is 0.9. So, need to you may leave your final answer in terms of equations. So, let me try to take only the first try to satisfy only the first constraint what should be the constraint on my  $\lambda$  such that the first constraint of probability  $D$  given  $a$  is 0.9 satisfied and the idea is very, very easy.

(Refer Slide Time: 15:53)

The image shows handwritten mathematical notes on a whiteboard. At the top, there is a formula:  $P(D|a) = \frac{e^{\lambda_1}}{Z}$ . Below this, two equations are shown:  $P(V|a) = \frac{1}{Z} [e^{0\lambda_1}] = \frac{1}{Z}$  and  $P(N|a) = \frac{1}{Z} e^0 = \frac{1}{Z}$ . To the right of these, another formula is written:  $P(D|a) = \frac{e^{\lambda_1}}{e^{\lambda_1+2}} = 0.9$ . A small logo in the top right corner of the whiteboard reads "CET IIT KGP".

You will just that like we found out probability D given cat, you will now try to find out probability D given a, and that will be  $e$  to the power lambda 1, yes because first feature is 1 for this case divided by Z. Now, what is Z? So, for finding out Z, I have to find out probability V given a, and probability N given a, now brought this probability V given a. So, again I will write 1 divide by Z into something  $e$  to the power lambda 1 times f 1 and so on. Now the word a is given only in the first feature, yes, first feature says the word is a and tag is D. So, the word is a is given; it is not given in feature 2, feature 3.

Now, what about the other features f 4, f 5, f 6, they all talk about words in V prime where word is not there. So, any of the feature cannot be one for this case, so all of them will be 0. So, it will be simply  $e$  to power 0, all the features are 0. Same happens with N given a, so they are 1 by Z, 1 by Z, and this tells me that probability D given a should be  $e$  to the power lambda 1 divided by  $e$  to the power lambda 1 plus 2, and there should be 0.9 to satisfy the constraint. And that will give me the value of lambda 1. Similarly, you will do for all the six cases and write down the questions. So, hope that gives you a good idea on how do we actually start building a maximum entropy classifier from (Refer Time: 17:50) in the data and how do we compute the probabilities.

(Refer Slide Time: 17:54)

The specific word and tag context available to a feature is

$$h_i = \{w_i, w_{i+1}, w_{i+2}, w_{i-1}, w_{i-2}, t_{i-1}, t_{i-2}\}$$

Example:  $f_j(h_i, t_i) = 1$  if  $\text{suffix}(w_i) = \text{"ing"}$  &  $t_i = \text{VBG}$

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lec 1

Now, I will take a very specific case. So, that is by the paper written in 1996 by Ratnaparakhi on using maximum entropy model for part of speech tagging. So, what I will show is that what kind of features they are used for part of speech tagging. So, now, this model is there now what is important to understand is that given a problem, what sort of features I should use to get a good performance, model will work well if the features that you have chosen you have chosen make sense for this particular problem. So, for part of speech tagging problem, what are the kind of features that he had used.

Now, given a new problem again you have to go back and find out what should be the interesting features you know now the format of features how you should define your features, but what are important features we will always depend on the particular problem that you are solving. So, as we had said in the maximum entropy model you choose a context around the word from where you can choose your features. So, in this case, what is the context they have chosen? They have chosen the current word, the next two words, the previous two words, and the tags given to the previous two words. And that is interesting how do you use this kind of features at the time of decoding when given a sequence you are trying to find out the tags. Because you do not know what is the best tag for the previous word. So, we will look at this problem, when we talk about the search algorithm the beam search algorithm that how do we make use of previous tag and previous tag here are to be assigned that are not yet assigned fully, but here use always features in all this context for defining this features.

Now, what are the form of features? So, yes just to give you another example, features will depend on the history or the context and the current tag. So, it is one if the current word has a suffix of ing and the tag is VBG that is one kind of feature; the suffix of the current word and the tag. So, what are the other features?

(Refer Slide Time: 20:02)

Example Features		
Condition	Features	
$w_i$ is not rare	$w_i = X$	$\& t_i = T$
$w_i$ is rare	$X$ is prefix of $w_i$ , $ X  \leq 4$	$\& t_i = T$
	$X$ is suffix of $w_i$ , $ X  \leq 4$	$\& t_i = T$
	$w_i$ contains number	$\& t_i = T$
	$w_i$ contains uppercase character	$\& t_i = T$
	$w_i$ contains hyphen	$\& t_i = T$
$\forall w_i$	$t_{i-1} = X$	$\& t_i = T$
	$t_{i-2}t_{i-1} = XY$	$\& t_i = T$
	$w_{i-1} = X$	$\& t_i = T$
	$w_{i-2} = X$	$\& t_i = T$
	$w_{i+1} = X$	$\& t_i = T$
	$w_{i+2} = X$	$\& t_i = T$

Navigation icons: back, forward, search, etc.

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lecture 4      4 / 8

So, what is one interesting thing here? We divided it features into three parts; some features are only for those words that are not rare that a very, very common. Second sort of features are those words that are rare. Why does it define separate features for the words that are rare? Because these words may not be seen again in my training data, but I might see some other real words, so can I use some of the properties of these words instead of using the word directly? So, further words that are not rare, they occur many times uses a word directly  $w_i$  is equal to  $x$ , and its current tag this is the form of feature. But if the word is rare, he does not use the word directly, he says  $x$  is a prefix of  $w_i$  length of  $x$  is less than equal to 4. So, he uses all the prefixes of the word is starting from 1, 2, 3 and 4, same for suffix.

Now, how will that be helpful using the suffix? So, for example, I have words that ends in ed, but overall the word is rare. So, can I use the fact that the word ends in ed ok that is why we are using the suffix here similarly for prefix, I can have something like in for making the opposites for an adjectives and so on. So, this can be captured by using this kind of features. Whether the word contains a number or in uppercase character or in hyphen, this is for rare words. Now, he just some other features for all the words. So, what are the features? The

previous tag is x and the current tag is t, this is very, very generic feature. The previous two tags are x and y, current tag is t; previous word is x and current tag is t; previous-to-previous word is x, current tag is t then similarly for the next for the next, next words.

(Refer Slide Time: 22:19)

Word:	the	stories	about	well-heeled	communities	and	developers
Tag:	DT	NNS	IN	JJ	NNS	CC	NNS
Position:	1	2	3	4	5	6	7
$w_i = \text{about}$	& $t_i = \text{IN}$				$w_{i-1} = \text{about}$	& $t_i = \text{JJ}$	
$w_{i-1} = \text{stories}$	& $t_i = \text{IN}$				$w_{i-2} = \text{stories}$	& $t_i = \text{JJ}$	
$w_{i-2} = \text{the}$	& $t_i = \text{IN}$				$w_{i+1} = \text{communities}$	& $t_i = \text{JJ}$	
$w_{i+1} = \text{well-heeled}$	& $t_i = \text{IN}$				$w_{i+2} = \text{and}$	& $t_i = \text{JJ}$	
$w_{i+2} = \text{communities}$	& $t_i = \text{IN}$				$t_{i-1} = \text{IN}$	& $t_i = \text{JJ}$	
$t_{i-1} = \text{NNS}$	& $t_i = \text{IN}$				$t_{i-2}t_{i-1} = \text{NNS IN}$	& $t_i = \text{JJ}$	
$t_{i-2}t_{i-1} = \text{DT NNS}$	& $t_i = \text{IN}$				$\text{prefix}(w_i) = w$	& $t_i = \text{JJ}$	
					$\text{prefix}(w_i) = we$	& $t_i = \text{JJ}$	
					$\text{prefix}(w_i) = wel$	& $t_i = \text{JJ}$	
					$\text{prefix}(w_i) = well$	& $t_i = \text{JJ}$	
					$\text{suffix}(w_i) = d$	& $t_i = \text{JJ}$	
					$\text{suffix}(w_i) = ed$	& $t_i = \text{JJ}$	
					$\text{suffix}(w_i) = led$	& $t_i = \text{JJ}$	
					$\text{suffix}(w_i) = eled$	& $t_i = \text{JJ}$	
					$w_i \text{ contains hyphen}$	& $t_i = \text{JJ}$	

So, now let us see what will be the form these features will take. So, this is the generic form of the features. Now, let us take a simple context and see what are the forms these features are going to take. So, this is one context, I have a sentence the stories about well heeled communities and developers, and the tags are DT, NNS all the part of speech tags are given to me. So, this is for my label data.

Now, what are the values these features will take? So, firstly, I will differentiate between the rare words and not so rare words. So, what can be the rare words here? For example, the word well-heeled here is a rare word; and other words like stories, communities, developers are not rare. So, for the words that are not rare what feature am I using the current word and the tag? So, my feature could be the word is a stories, and tag is NNS and so on. So, for the word well heeled I will use the suffix and prefix. So, W is a prefix of the word and tag is JJ; W is a prefix and tag is JJ; similarly ed is a suffix and tag is JJ and so on. And then there will be some features for all the words like for about and well-heeled, what would be the feature the previous tag is IN and the current tag is JJ. The previous two tags are NNS and IN, and the current is JJ. The previous words or the word stories and well-heeled about and well-

heeled and so on, so you can use all these features. So, here are all these features. So, this is prefix and suffix for the rare word well-heeled.

And there are other things like w i contains hyphen that is in the case of well heeled and tag is JJ. So, now that is how you will write down all your features from your data. And you will learn your parameters lambdas and all; and at run time you will use again these features in the parameter to finding the probability of the tags. So, hope this gives you some idea, and how can you choose your features for a given task.

(Refer Slide Time: 24:37)

**Search Algorithm**

**Conditional Probability**

Given a sentence  $\{w_1, \dots, w_n\}$ , a tag sequence candidate  $\{t_1, \dots, t_n\}$  has conditional probability:

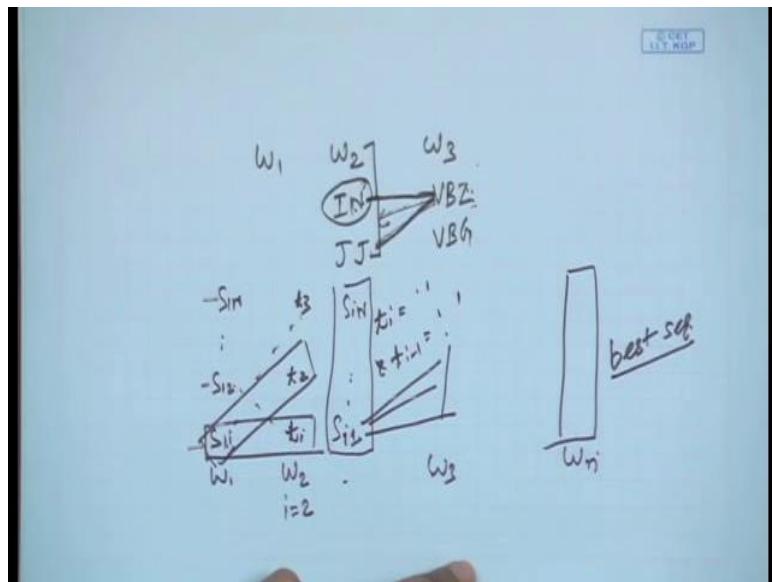
$$P(t_1, \dots, t_n | w_1, \dots, w_n) = \prod_{i=1}^n p(t_i | x_i)$$

A *Tag Dictionary* is used, which, for each known word, lists the tags that it has appeared with in the training set.

Pawan Goyal (IIT Kharagpur)      POS Tagging      Week 4, Lec 1

Now, let me go to the search algorithm for this maximum model. So, this we had discussed earlier also, but we did not discuss in full details. Some given a sentence w 1 to w n these are the words, and I want to find out the probability of tag sequence t 1 to t n. And we said that we can write it in this form multiplication of probability t i given x i, x i is the context for each individual word. Now, the problem here is and you can use some tag dictionary that says for each individual word what are the possible tags similar to what we did in the case of Viterbi decoding for HMM. Now, what is one particular problem in this case? So, as such it looks as if you can do it independently for each word right; for the first word I can find out what is the best tag multiplied with best tag for the second word and so on. Now, what is one problem with this approach let me just give you one simple example.

(Refer Slide Time: 25:36)



So, this is a hypothetical example. So, let us say that we have word 1, word 2, word 3. And word 2 can take two tags IN and JJ; and word 3 can take lets to keep it simple that it can take like VBZ and maybe something else like VBG maybe they may not be possible, but just a hypothetical case. Now, suppose by using this algorithm or doing it independently, we found out that for w 2, IN give some probability and JJ gives some probability. Now, in w 3 what might what could happen some of the features might depend on the previous assigned tag. So, for VBZ you might want to use what is the tag assigned in the previous word so that means, choosing the probabilities the best probability at this point may not be the best.

So, choosing the tag at this point may not be the best because at the next step you are using the tags and the probabilities can change this can happen that IN is giving the better probability at this point, but when combined with VBZ, JJ gets a better probability this can happen. So that means, I can find out the probabilities, but I have to use the probabilities and the tags in the next step also. So, this is not fully independent. I cannot just choose IN and forget about it I have to remember what is the probability next time I will choose IN, VBZ, JJ VBZ and try to compute the probability for VBZ multiplied by the previous probabilities and that is how exactly we do that I will show you in the beam search algorithm.

(Refer Slide Time: 27:34)

**Search Algorithm**

Let  $W = \{w_1, \dots, w_n\}$  be a test sentence,  $s_{ij}$  be the  $j$ th highest probability tag sequence up to and including word  $w_i$ .

**Search description**

- Generate tags for  $w_1$ , find top  $N$ , set  $s_{1j}, 1 \leq j \leq N$ , accordingly.
- Initialize  $i = 2$ 
  - Initialize  $j = 1$
  - Generate tags for  $w_i$ , given  $s_{(i-1)j}$  as previous tag context, and append each tag to  $s_{(i-1)j}$  to make a new sequence
  - $j = j + 1$ , repeat if  $j \leq N$
- Find  $N$  highest probability sequences generated by above loop, set  $s_{ij}$  accordingly
- $i = i + 1$ , repeat if  $i \leq n$
- Return highest probability sequence  $s_{n1}$

Pawan Goyal (IIT Kharagpur) POS Tagging Week 4, L2

So, what is the beam search algorithm? So, your test sentences containing words like  $w_1$  to  $w_n$  and let say that for the  $i$ th word as  $s_{ij}$  denotes the  $j$ th highest probability tag. This is probability tag sequence up to the word  $w_i$ . Now, how does the algorithm work? So, you have the words  $w_1$  to  $w_n$ . For word  $w_1$ , you have  $s_{11}, s_{12}$  up to  $s_{1N}$  top and highest probability tags. So, we will come to the probabilities and this is easy again you will use the features here which feature is 1, which feature is 0 then you will do the normalization you will compute all these probabilities for the each of these tags.

Now, the important thing is how do you go to  $w_2$ . So, one thing to understand is that you can probably not keep all the possible tag sequences, because assume that each word can take on an average  $k$  tags. So, if you have  $N$ -words, the tag would be of the order of  $k$  to the power  $n$ , yes  $k$  to the power. So, this will be exponential in the length of the sentence. So, you cannot keep all the tag sequence. So, you might have to make the choice of the best sequence is available at this point that is what we do in bean search. So, at each point, I will select what is the top set up capital  $N$  sequences at this point and how do we how do we select that, so that is the next part of the algorithm.

So, initialize  $i$  is equal to 2. So, now you have to second word, generate tags for  $w_i$  given  $s_{i-1j}$  as previous tag context and append each tag to  $s_{i-1j}$  to make a new sequence and then continue for all the tags. Now, what does that mean? Initialize  $i$  is equal to 2 that is your second word. Now, generic tags for  $w_i$  given  $s_{i-1j}$  as the previous tag context

what is  $i$  minus  $j$ ,  $i$  minus 1 is  $s_1$  and  $j$ . So, let me take this as the previous context. And suppose this has again some tags like I will just write some tags  $t_1, t_2, t_3$ . So,  $s_1$  and  $t_1$  becomes the first sequence;  $s_1$  and  $t_2$  becomes the second sequence; and  $s_1 s_2 t_3$  becomes the third sequence. Similarly,  $s_1 s_2 t_1$  becomes forth sequence and so on. So, in this case, there will be three times  $n$  sequences. So, this you can take in general case this may not be one tag this will be a sequence up to this point. So, we have all the first sequence at this point.

Now, for each of the sequence, you know what are the previous tags that have been assigned? Yes, so I know what are tags assigned at this point. So, I can use all the features. So, if a feature says  $t_i$  is equal to something and  $t_{i-1}$  is equal to something else. So, I can check if  $t_i$  matches this, and the previous tag matches this then only it will do 1, otherwise 0, because I am choosing the context, I can compute all these features. I can compute these features; I can compute the probability for all the sequences. And now what I will do I will select at this point what are the  $s_1$  to  $s_N$ , what are top end sequences at this point. And again I will go to the third word, again its tags combine with all the sequences and choose top end from here. So, what will happen use your space will not do exponentially, you will always have top end sequence is at any given point.

So, finally, when you go to the final word, you will get the top end probabilities or all the probabilities, and you will select the best sequence. And now because the sixth is sequence contains the tag from by starting, you will find out the part of speech tag just from the choosing the best sequence at the last word and that is your algorithm. So, find  $N$  highest probability sequences by above loop set this accordingly and repeat if  $i$  less than equal to  $n$ . And return to highest probability sequences  $x_n$  for the last word, and this is your maximum entropy this you can call as maximum entropy Markov model. Instead of maximum entropy model is generate classifier that can do for each word individually, but when you are trying to applied for a sequence labeling problem for a sequel, you use this maximum entropy Markov model. So this was maximum entropy Markov model for you.

Now, in the next lecture, I will take one simple example of this search, and I will quickly cover the model of conditional in the fields that is one of the state of the arts in sequence labeling task. And what we will show, there is some problem with the maximum entropy model that condition of fields try to handle.

Thank you, I will see you in the next class.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 21**  
**Conditional Random Fields**

Welcome back for the final lecture of this week. So, we are talking about the problem of part of speech tagging and the methods that we are discussing our generic methods for any sequence labeling task. So, input is sequence like here sequence of the words; and output is again a sequence where for each word I going to predict what is a corresponding part of speech tags. So, we talked about hidden Markov models and also Maxent model. So, in general in Maxent model, we talked about how do you we use that simple classifier for a sequence labeling task.

So, we will call that the maximum entropy Markov model. And the formulation was very easy that is if predict the tag for each word, and then multiply the probabilities for the whole sequence. There was one problem though that because we need the tag of the previous word in certain features to assign the tag at this particular word, we will need to use beam search algorithm and we also discuss; what is the beam search algorithm.

So, what I will do in this lecture I will take a take an example there we will see how to use beam search algorithm and then I briefly discuss what are condition random fields and how are the different from maximum entropy models. So, condition random fields again are a very vast topic, we will not cover fully. We will only give you the hint that once you know Maxent or MEMA models what are condition random fields how are they different from those.

(Refer Slide Time: 01:56)

**Practice Question**

Suppose you want to use a MaxEnt tagger to tag the sentence, "the light book". We know that the top 2 POS tags for the words *me*, *light* and *book* are {*Der*, *Noun*}, {*Verb*, *Adj*} and {*Verb*, *Noun*}, respectively.<sup>22</sup> Assume that the MaxEnt model uses the following history  $h_t$  (context) for a word  $w_t$ :

$$h_t = \{w_{t-1}, w_{t+1}, t_{t-1}\}$$

where  $w_{t-1}$  and  $w_{t+1}$  correspond to the previous and next words and  $t_{t-1}$  corresponds to the tag of the previous word. Accordingly, the following features are being used by the MaxEnt model:

- $f_1: t_{t-1} = \text{Der}$  and  $t_t = \text{Adj}$
- $f_2: t_{t-1} = \text{Noun}$  and  $t_t = \text{Verb}$
- $f_3: t_{t-1} = \text{Adj}$  and  $t_t = \text{Noun}$
- $f_4: w_{t-1} = \text{the}$  and  $t_t = \text{Adj}$
- $f_5: w_{t-1} = \text{the} \& w_{t+1} = \text{book}$  and  $t_t = \text{Adj}$
- $f_6: w_{t-1} = \text{light}$  and  $t_t = \text{Noun}$
- $f_7: w_{t-1} = \text{light}$  and  $t_t = \text{Verb}$
- $f_8: w_{t-1} = \text{NULL}$  and  $t_t = \text{Noun}$

Assume that each feature has a uniform weight of 1.0.

Use Beam search algorithm with a beam-size of 2 to identify the highest probability tag sequence for the sentence.

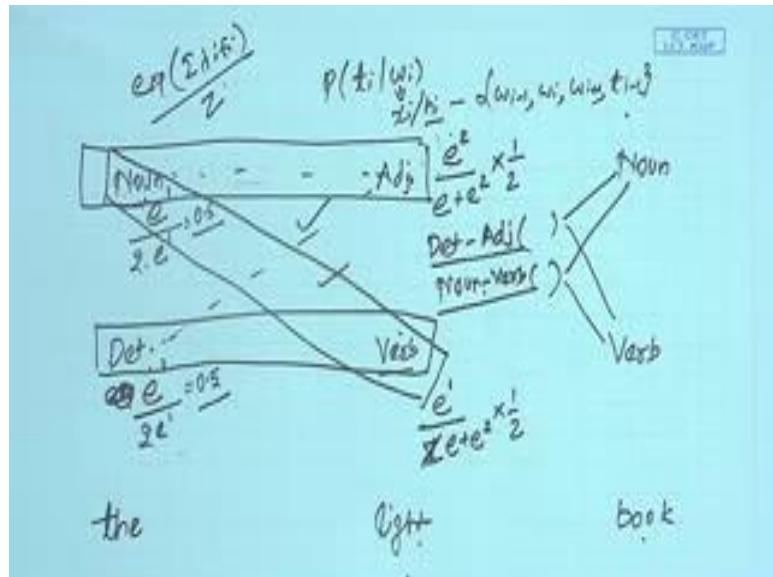
So, we are starting with a practice question. So, here so we are having the same sentence the light book; and you are given that for all the three words the, light and book, the top two excel determine and noun, verb, adjective and verb noun. Now, you want to use your MEMA model. So, for the given tag or given word  $w_i$ , you use particular context what is a context here the previous word next word and the tag given to the previous word this is the context so that means, all your features we will be defined over this context. So, here is a example, we have showing some sample like you have given a different features. And now we have to use the beam search algorithm to find out what will be the appropriate tag sequence for the sentence.

So, what are the features giving? So, features are simple like the previous tag, the tag given to the previous word is determiner, and current tag is adjective; previous tag is noun, current tag is verb; previous tag is adjective, current tag is noun; previous word is the current tag is adjective and so on. Then you also have feature like the next word is light, current tag is determiner; previous word is null and current tag is noun. What it means is that this  $i$ th word will be starting a sentence that is why the previous word will be null.

So, now, you are given these features and for simplicity you are should given that each feature has a uniform weight of 1. Now, your task is to use beam search algorithm with the beam size of 2. What do you mean by a beam size of 2, at any given point, you will keep only the top two highest probability tag sequence, and everything else you will forget. So, at any

point you will know; what are the top two tag sequences till this point. And overall you have to find the highest probability tag sequence for the sentence that is the light book.

(Refer Slide Time: 03:56)



So, let us see how do we solve this. So, we are having three words in the sentence the, light and book. The word has two text two possible texts it can be either a determiner or a noun. The word light can be a verb or an adjective; and the word book can be a verb or a noun. So, now how do we start you to find out probability tag  $i$  given  $w_i$  or instead of  $w_i$ , let me write the context  $x_i$  or  $h_i$  give different notations for that. So, here context is  $w_i$  minus 1  $w_i$   $w_i$  plus 1 and  $t$  is tag. And what is the formulation of this tag  $i$  given the current word current here, it will be exponent summation lambda  $i f_i$  and feature we know is a function of input and the tag divide by  $Z$ . And  $Z$  is nothing but a normalization constant, so that all the text probability adds up to 1.

So, let us try to do that. When the tag is determiner word is the. So, let we just write down exponent of summation lambda  $i f_i$  and lambda  $i$  is lambda  $h$  is 1 here in this problem. So, it will be simply exponent summation over  $f_i$ . So, what features are 1 and what features are 0. So, for this word, so everywhere where we need the previous tag or the previous word should be 0, because this is the start of the sentence. So, I do not have any previous word tag or any previous word. So, all these features value will become 0.

Now, what is the feature that will become 1 what should be this one, this needs the next word  $i$  plus 1 word is light and the current tag is determiner is that will that 1. So, if you see here

current tag is determiner on the next word is light. So, it could be 1 that is for feature f I. So, it is 1, so I will say it is exponent or let me write e to the power 1, this lambda is 1 divided by z, let me find out that Z. Let me find out the value for noun. So, again similar to this one all of features from f 1 to f 6 will become 0, because you do not know what is a previous word or previous tag this feature could have become one, but the current tag is not determiner, but noun. So, this will also be 0. This feature previous word is null yes that is true is a start of the sentence, and the current tag is noun, this will also become 1.

So, this is the only features that become the 1 for noun. So, this is e to the power 1. And what is the normalization this plus this. So, 2 times e to the power divided by 2 times e to the power 1. So, both will become 0.5. So, at this point all the two tags have we both the tags has probability 0.5. And any of because I am using a beam size of 2, I will have to keep both these tags. So, now I keep both this tags with probability 0.5, 0.5.

Now let us go to the next word light. Now again so I have to use probability of verb given this history; now this history is previous word, current word, next word and the previous tag. So, now, when I going about talking about verb, if I have to compute these features, I need to know what is the previous tag, here it can be a either determiner or noun. So, that is why I need talk in terms of the sequences. So, here I have one sequence, determiner-noun, determiner-verb, and second sequence with noun-verb. So, we have to take both the sequences separately in compute the probability. Similarly, I will have to do the same for adjective.

So, let us try do that for one sequence. So, noun-verb sequence. So, what will be the probability of tag i given the word. So, this we can write here. So, let me write only this summation lambda i f i part. So, this will be summation lambda i f i. So, let us go to the features here, so noun-verb. So, let us go to the features first features previous tag is determiner no previous tag is noun here this is 0; previous tag is noun, yes and the current tag is verb this is this could be 1, so f 2 is 1. F 3 -0, because previous tag is not a adjective. F 4 previous tag is the noun, yes previous word is the current tag is adjective, no it is verb then this will be also not correct, because the current tag is not a adjective, but verb; previous word is light noun, next word is light noun, previous word is null noun. So, only f 2 is 1. So, this will be for this sequence it will be e to the power 1 divided by the normalization Z.

Now what will this normalization depend on this will depend on from all everywhere where this context is taken what are the probabilities. So, I have to compute the probability for this one also to find out this  $Z$ . So, I know this probability, I know this function this function and I will normalize them to add to 1. So, what will be for the function for noun and adjective I just try that from the features again. Previous tag is determiner gone, previous tag is noun, but current tag is verb noun, previous tag is adjective noun, previous word is the yes and tag is which adjective yes. So, this will become one of four will become 1. Previous word is the yes, next word is book yes, current tag is adjective this will also become 1 all three will be 0. So, now, this will become  $e$  to the power 2 divide by  $Z$ ; and  $Z$ , I can write as  $e$  plus  $e$  square; same here  $e$  plus  $e$  square.

So, now I know the probability of getting verb at this position given the previous tag is noun; and adjective at this position given the previous tag is noun. But what is the probability of this whole sequence it will be multiplied by the probability of getting noun that is half, similarly here half. So, this is the probability of selecting this sequence similarly I will compute the probability of selecting this and this normalized them multiplied by 0.5. So, now, I will get the probability for four sequences, so that is noun-verb, determiner-verb, determiner-adjective, and noun adjective, I have the probability of four sequence of this system. And then I will select only top two from there.

So, suppose the top two could be say noun-verb and determiner-adjective. So, what will happen now, for the next step, I will consider only say determiner-adjective and noun-verb, and I will know also their probabilities. Then I will take each individual as a history and see determiner-adjective then noun, determiner-adjective then verb normalize the probability. Similarly noun-verb - noun, noun-verb - verb normalize the probability accordingly multiply this. So, again you will have four sequences here, you will have probability for all four sequences and take the one that is having the highest probability and that will be your final tag sequence in this example.

So, I hope the idea is clear I am not solving this fully, but I am I will encourage you that you do it on your own, and see that you can find out what is the appropriate sequence using the MMA model. So, this was how to use beam search algorithm for MMA model. Now I will just talk briefly with what is the problem with this, what is the single problem with this maximum entropy model then? So, let we have to think about condition random fields.

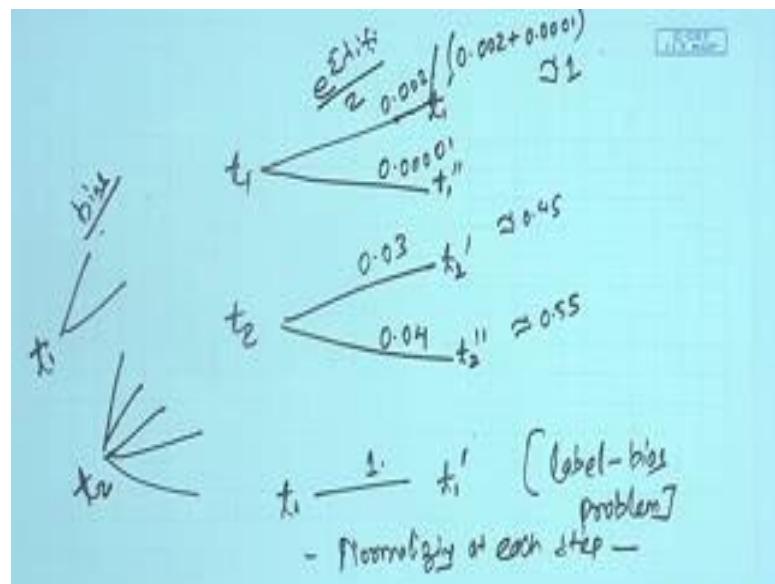
(Refer Slide Time: 13:30)

*Per-state normalization*  
All the mass that arrives at a state must be distributed among the possible successor states

*This gives a 'label bias' problem*  
Let's see the intuition (on paper)

So, in maximum entropy model, we do a per state normalization that is all the mass that arrives at a state must be distributed among the possible successor states, and this is giving rise to a label bias problem. So, let us see; what is the intuition. So, what do I mean by this. So, let me take the same example. So, first let me take the same example to explain what do I mean by normalization at each state. So, take this one. So, you are computing noun-verb and noun-adjective. And you had the features like  $e^{\text{square}}$  and  $e^{\text{1}}$ , but you were normalizing them by so you are normalizing them such that these two add up to 1. Same thing, you will do with determiner; you will make sure that these two add up to 1. So, you are normalizing at each state, so why will that a problem.

(Refer Slide Time: 14:40)



So, let me just take a hypothetical example. Suppose, in your maximum entropy model you were having a tag  $t_1$  and tag  $t_2$  at any given point. So, next point, suppose from  $t_1$ , you can go to two different tags  $t_1'$  and  $t_1''$ . And from  $t_2$  again you can go to  $t_2'$  and  $t_2''$ ; they may be same, they may not be same. Now, how do you compute this probability, it will be  $e^{\sum \lambda_i f_i}$  divide by  $z$ ;  $z$  is nothing but the addition of these two. Same here now this is what is the importance for how many features you were having and so on.

Suppose, for a particular choice of these tags, it happens that in one of the branch, they are having this value has 0.002; and this value has 0.0001 or let us say even much smaller value. And this branch is having value of 0.03, 0.04. So, what will happen now this is not normalized values. So, it tells that this summation  $\lambda_i f_i$  is getting a higher score in these two cases; and lower score in these two cases. But because you are doing for normalization, you will divide by 0.002 plus 0.0001 and that will be closed to say very close to 1, 0.98 or something; on the other hand this will be closed to 0.45 or 0.55.

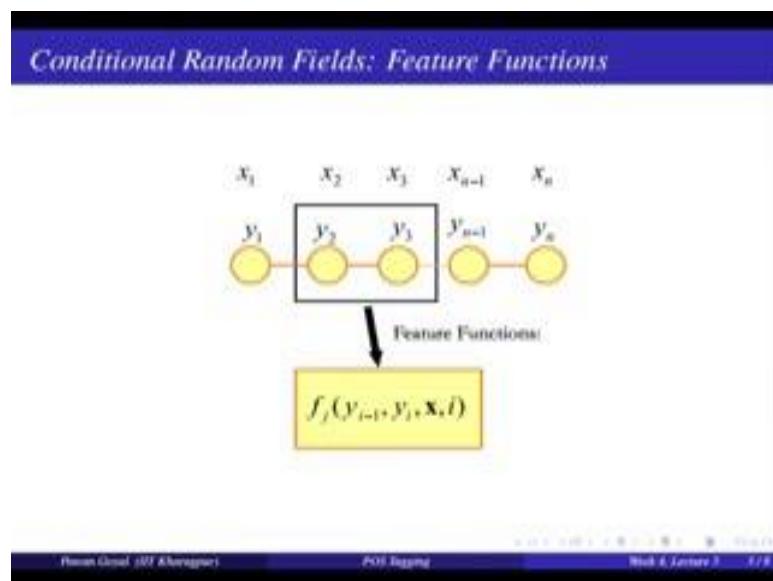
So, what is happening here even though this probability was low when I normalized this became very this became very high that is one particular problem. On the other hand, suppose that from  $t_1$ , there was no possibility of going to  $t_1''$ . So, there was only one tag possible. So, you will in independent of the context, it will always get a probability of 1, because you have to normalize at each state that is if from  $t_1$ , I can only go to one tag  $t_1$ .

prime and everything else has a probability of 0, because if normalization this will become 1. So, we multiply this probability t 1 independent of the context and this is called as the label bias problem. And this comes because you are normalizing at each step that is one problem with the maximum entropy Markov models.

So, let us see how we avoid this problem in condition random fields. So, I hope this problem is clear that you are doing normalization at easy step at each state and that is giving some bias towards those states that are having few transitions then the (Refer Time: 18:10) having more number of transitions. So, let me just telling one thing. So, suppose I have two different tags from t 1, you have two possible transitions; from t 2, you have five possible transitions. What will happen these will get there will be a bias towards choosing this state because this will be normalized and one of this will get a higher value and this may not happen here because there are five possible transitions. So, this gets a bias and that is not what is idle.

So, how do we avoid this problem and conditional random fields? So, conditional random fields are undirected graphical models, and while there are many variations of conditional random fields, so there is a generic structure, we will look at the linear generic structure.

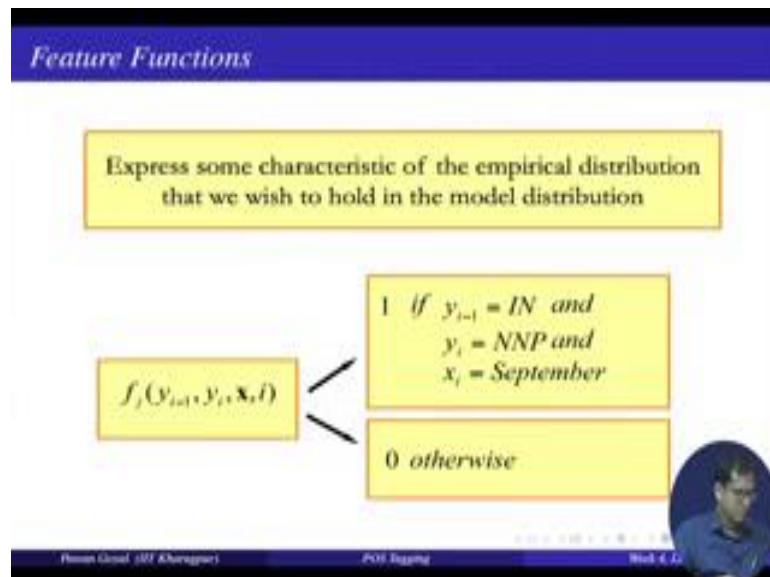
(Refer Slide Time: 19:08)



So, here we are seen the linear generic structure of conditional random fields. So, again like in the previous case, you are having a sequence  $x_1$  to  $x_n$  and these are the tags  $y_1$  to  $y_n$  assign to these tag. Now how they are so in what way they are very similar to maximum entropy model they are similar in the sense that we they use the same for the feature

functions. So, what you are seeing here, for the  $i$ th point the feature function, so suppose  $i$  is equal to 3 would be a function over the previous tag  $y_2$ , current tag  $y_3$ , the whole the input you can take any number of words before and after  $x_3$  and this is  $i$ th index. So, feature functions are again function of the input current tag and previous tag, this is in the linear generic structure.

(Refer Slide Time: 20:17)



So, conditional random fields are like factor graphs. So, what happens the probability of each node will depending on only its neighbor. So, and you can use the same sort of features. So, like this is what we had discuss in maximum entropy model also. So, I have a feature that is 1, if previous tag is I N current tag is NNP and the current word is September and 0 otherwise. So, we will see there are very similar sort functions that we are using in Maxent. So, so they are same in as Maxent in that sense, but how they are different.

(Refer Slide Time: 20:45)

*Conditional Random Fields: Distribution*

Label sequence modelled as a normalized product of feature functions:

$$P(y | x, \lambda) = \frac{1}{Z(x)} \exp \sum_{i=1}^n \sum_j \lambda_j f_j(y_{i+1}, y_i, x_i, i)$$
$$Z(x) = \sum_{y \in \mathcal{Y}} \sum_{i=1}^n \sum_j \lambda_j f_j(y_{i+1}, y_i, x_i, i)$$


Powerpoint (MF Kherayev)  
PDF Slides  
Week 4

So, a difference comes in how the normalization is done. In Maxent model or maximum entropy Markov model, we are doing normalization for each state, so that is at easy state if I have multiple transitions, I will make sure that the probability for them at adds up to 1. This does not happen in conditional random fields. So, we will compute the features expectations of feature values for each possible transition, whole sequence and then I will do normalization, so that we can see from the probability function here. So,  $y$  is a whole sequence  $y_1$  to  $y_n$  given a current the current input sequence  $x_1$  to  $x_n$  and  $\lambda$  is the feature weights that you will learn. And this is 1 by  $Z$ -  $x$  say a single normalization parameter exponent summation over  $i$  is equal to 1 to  $n$  summation  $j$   $\lambda_j f_j$ . So, let us try to understand quickly what this function means.

(Refer Slide Time: 21:52)

$$P(y|x, \lambda) = \frac{1}{Z(x)} \exp \left( \sum_{j=1}^n \lambda_j f_j(\quad) \right) \checkmark$$

$y = y_1 \dots y_n$  many such seq. —

$x = x_1 \dots x_n$

$\sum_{y \in Y} \exp \left( \quad \right) e^x \cdot e^{y_1} \dots e^{y_n}$

$$\prod_{i=1}^n P(y_i|x_i) = \prod_{i=1}^n \exp \left( \sum_j \lambda_j f_j \right)$$

$$= \exp \left( \sum_{i=1}^n \sum_j \lambda_j f_j \right)$$

So, here having 1 up on Z x exponent summation i is equal to 1 to n summation over j lambda j f j and say f j is a jth feature. And this is probability y given x lambda, lambda are the feature of x. Now, let us try to understand this. What do you mean by y, y is a sequence y 1 to y n; and x is the input sequence, and there are many such sequences possible, so this is a probability for a given sequences. And this Z is a normalization that is done over all, so that is a summation over all the sequences. So, we can write it as, this will be summation over all the sequence. I have this value for only sequence y current sequence y, I will add it over all the sequence that will give me Z, so summation over all y possible y exp and whatever was in set.

So, now how do we get this equation? So, remember this equation summation j lambda j f j that is for particular tag given at the ith position. So, I have exponent summation j lambda j f j that is probability of a tag y i given x i; x i can be all history at the given point divided by Z was there, but forget about the Z term. Now, here we are computing probability for the whole sequence y i, i is equal to 1 to n. So, so this will be multiplication. So, multiply i is equal to 1 to n. So, multiply i is equal to 1 to n. Now, if I multiplying multiple exponent this is like, so for example, e to the power x times e to the power y becomes e to the power x plus y. So, that is multiplication of all the exponent is nothing but like summation of what is inside exponent summation i is equal to 1 to n summation j lambda j f j and that is the function a. And Z x is normalization over all such transitions all such sequences.

So, you are not doing normalization here. So, what happens in Maxent you are doing normalization here? So, for a given  $i$ , you make sure that everything adds up to one and that is not being done here. You are not making sure that at each  $i$  all the tags will the probability for the text will add up to 1. Now, you are doing a normalization only in the end, I know the probability or something that is proportion to probability for each tag sequence and then I will normalize everything by this  $z(x)$  and that is way this avoids the label bias problem. So, this is the particular function that is using conditional random fields.

(Refer Slide Time: 25:24)

The slide has a blue header bar with the text 'CRFs' in white. The main content area contains a bulleted list:

- Have the advantages of MEMMs but avoid the label bias problem
- CRFs are globally normalized, whereas MEMMs are locally normalized.
- Widely used and applied. CRFs have been (close to) state-of-the-art in many sequence labeling tasks.

At the bottom of the slide, there is a video player interface with the following labels: 'Praveen Choudhary (IIT Kharagpur)', 'PDF Slides', and 'Week 4, Lec 1'.

So, what we can see that. So, conditional random fields have the advantage of maximum entropy Markov model, they use a same sort of features, the kind of model is very, very similar to maximum entropy Markov model. But they avoid the label bias problem. So CRFs are globally normalized, whereas MEMMs are locally normalized, so that we had discussed. And they are very widely used and applied for many, many sequence labeling task. So, they were very closed to the state of the art models for many of these sequence labeling task. So, whatever sequence labeling task that comes your mind, so starting from part of speech tagging to name and recognition.

So, there you can apply a conditional random field model and lots of libraries are also available. So, (Refer Time: 26:13) one particular app that is very popular, and then there are CRA plus plus and many other libraries that you can use. What is important is that you

understand, what is the sort of features that that you need to use and then the model will help you to train your own CRF.

So, this ends our discussion on part of speech tagging we did discuss a lot about what will be the models for sequence tagging. So, from the next week, we will start discussion on syntax that how do we find out what are the words, what are the word arrangement in a sentence and how do we group them in various sort of phrases. So, I will see you next week.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 22**  
**Syntax introduction**

Welcome to the 5th week of this course. So, we have so in this week, we are going, we will be moving up, the hierarchy of solving this language processing tasks. So, we had started with discussing about how do we handle individual words? How do we use, how do we make use of the word or information using language models and then how do we assign various grammatical categories; a part of speech categories to the words that is what we are doing till now.

Now, we will go to next level where we will try to, see can we arrange these words in certain groups. So, this is what we study in this topic of syntax. So, how are the words being arranged together and what can we; how we can automatically find out this arrangement given a new sentence? This will be the topic of parsing that we will discuss in this week.

What is syntax? So, in general syntax refers to the way the words are arranged together and also you will see, what is the relationship between various words and the word groups that is what we will talk about in syntax, so, when we were talking about language models, we had discussed what is the importance of modeling word order. So, which words occur after what are the words? How we can make use of that in assigning probability for a sentence or finding out the next word in completion task or in spelling correction task, similarly when we were talking about part of speech task, we saw what are the grammatical categories, part of speech categories, so this defines in one sort of equivalence class for words that is all these words behave like they are verbs behave in some equivalent manner, similarly all these words are nouns, they behaving in some equivalent manner, similarly all these are adjectives. So, you are defining some equivalent classes.

(Refer Slide Time: 02:25)

The slide has a blue header bar with the title "What is Syntax?". Below the title is a bulleted list of four points:

- Refers to the way words are arranged together, and the relationship between them.
- Language Models:** Importance of modeling word order
- POS categories:** An equivalence class for words
- More complex notions: constituency, grammatical relations, subcategorization etc.

Below the list is a Yoda meme with the text:  
SYNTAX THE ONLY PART OF LANGUAGE  
IT IS NOT  
BUT IMPORTANT IT IS

At the bottom of the slide, there is a navigation bar with icons and text: "Pinwan Goyal (IIT Kharagpur)" on the left, "Syntax" in the center, and "Week 5: Lec 1" on the right. There is also a small circular portrait of a man in the bottom right corner.

Now, in syntax we will find out some more complex notions like word is constituency, what are different grammatical relations among the words? What groups or what words are grouped together in constituency and sub categorization etcetera also included in this in this topic and yeah just to make you understand, what is this notion of syntax? So, you can relate to that if you see this particular meme. So, I hope you remember this particular character from statics, this syntax the only part of the language it is not, but important is it is right. So, you can see the way words are arranged normally in the way we speak and how they arranged in this particular sentence that was the special characteristics of this character.

(Refer Slide Time: 03:17)

The slide has a blue header bar with the title "Syntax Tree: Example". Below the title is a parse tree diagram for the sentence "The man read this book".

```
graph TD; S --- NP1[NP]; S --- VP; NP1 --- Det1[Det]; NP1 --- NOM1[NOM]; Det1 --- The1[The]; NOM1 --- Noun1[Noun]; Noun1 --- man1[man]; VP --- Verb1[Verb]; VP --- NP2[NP]; Verb1 --- read1[read]; NP2 --- Det2[Det]; NP2 --- NOM2[NOM]; Det2 --- this1[this]; NOM2 --- Noun2[Noun]; Noun2 --- book1[book];
```

At the bottom of the slide, there is a navigation bar with icons and text: "Pinwan Goyal (IIT Kharagpur)" on the left, "Syntax" in the center, and "Week 5: Lec 1" on the right. There is also a small circular portrait of a man in the bottom right corner.

Now so what are we going to study in syntax. So, let me give you a simple example, So, I have the sentence, the man read this book. So, by syntax, we are trying we are trying to find out what are the various groups of word that are coming together for example, in part of speech text we found out that the word the as part of speech of determiner man as the part of speech of noun read as verb and. So, on now we are going one level up. So, now, we are saying this is determiner this is noun this is kind of nominal, but they both together make a phrase noun phrase.

I am saying the man is a noun phrase; similarly this book is also a noun phrase. So, there are 2 noun phrases here, the man and this book, but when the verb read or any verb comes before the noun phrase it makes a verb phrase.

All these 3 words act as a single unit of a verb phrase, there is a no such unit for man and read, there is a unit for read this book and the man and then I go up saying that a noun phrase and verb phrase are making the sentence and this gives me complete hierarchy structure of how the words arranged in the sentence the sentence is nothing, but a noun phrase and the verb phrase this noun phrase contains a determiner and noun verb phrase contains a verb and noun phrase which contains and so on. So, this is the complete hierarchy of the sentence that I come to know by this syntax tree and that is the; what is the topic of this week, how do we come up with such syntax tree for some sentences, what is the particular formulation that we will use.

(Refer Slide Time: 05:07)

*Defining the notions: Constituency*

**Constituent**  
A group of words acts as a single unit - phrases, clauses etc.

**Part of Speech - "Substitution Test"**  
The {sad, intelligent, green, fat, ...} one is in the corner.

**Constituency: Noun Phrase**

- Kermit the frog
- they
- December twenty-sixth
- the reason he is running for president

Let me start by defining some basic notions like what is constituency? So, in the last example we saw that a group of words; they act they acting as some single unit and you can call them as phrases clauses etcetera. So, in part of speech, we could have done this substitution test. So, I have this sentence, there is a fill in the blank, one in the room and I can fill in any adjective, the green one, the fat one and intelligent one, sad one, all that can be filled in.

So, I can fill in any word that belongs to that particular part of speech category, now here it will be a particular constituent, it will be a particular group of words that can behave similarly like here. So, all these 4 things; Kermit, the frog, the December, 26th the reason he running for president, all these are noun phrases and they can occur in a given context. So, now, for substitution test, you can substitute any of this these 4 noun phrases and yeah we will see an example where all these can come in the same in a very very similar context.

(Refer Slide Time: 06:25)

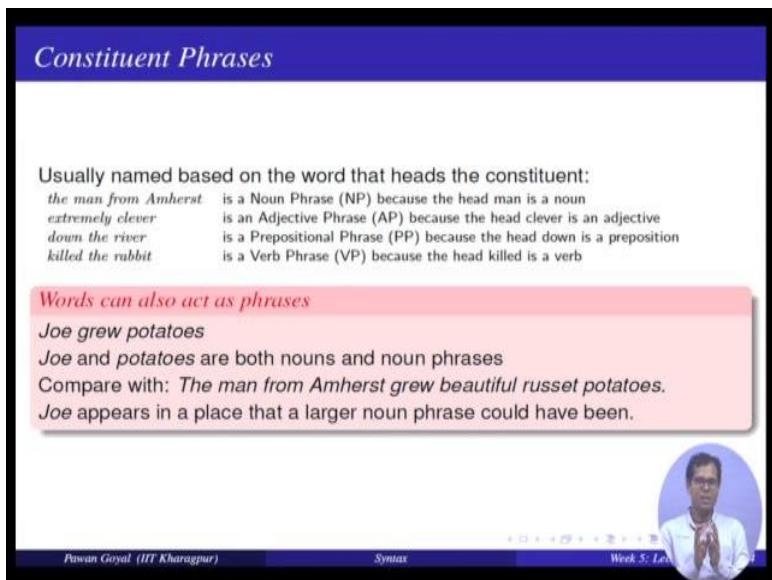
### Constituent Phrases

Usually named based on the word that heads the constituent:

<i>the man from Amherst</i>	is a Noun Phrase (NP) because the head man is a noun
<i>extremely clever</i>	is an Adjective Phrase (AP) because the head clever is an adjective
<i>down the river</i>	is a Prepositional Phrase (PP) because the head down is a preposition
<i>killed the rabbit</i>	is a Verb Phrase (VP) because the head killed is a verb

**Words can also act as phrases**

*Joe grew potatoes*  
*Joe* and *potatoes* are both nouns and noun phrases  
Compare with: *The man from Amherst grew beautiful russet potatoes.*  
*Joe* appears in a place that a larger noun phrase could have been.



How do we name these constituent phrases? So, last slide I was showing some noun phrases. So, why do we call them noun phrases or something else? So, usually the names are given based on the words that are heading these constituent, what is the head and usually speaking you can find the head by the word that can substituted for the whole thing, let me take the first example the man from Amherst, this is the phrase, they are 4 words, now which of the 4 word do you think can substitute the whole thing, now that can be used in the grammatical function of the complete unit and that will be man where the man from Amherst, the word

man can be used to denote the grammatical function of the whole unit. So, the head here is a noun; man. So, this will be called a noun phrase because the head man is a noun.

Similarly, extremely clever; the head here is clever; this is the adjective. So, this is called in adjective phrase down the river here; head is down preposition. So, this will be called in prepositional phrase, killed the rabbit; the head is killed, the word killed which is a verb. So, this is called verb phrase. So, like that we are, we defined, what are the constituent by taking what is the head of that phrase?

Now in general, words can also act as phrases. So, a phrase need not have always multiple heads, a single word can also be a phrase. So, let us take the simple example Joe grew potatoes. So, Joe itself it is a noun phrase, potato also a noun phrase, they are nouns, but also a noun phrases, in this case now compare the sentence with the man from Amherst grew beautiful russet potatoes. So, what do you see? So, instead of Joe I have substituted the man from Amherst, a complete a 4 word unit that is again a noun phrase and beautiful russet potatoes instead of potatoes they are still noun phrases. So, what happens in the sentence? Joe appears in a place where you could probably put a larger noun phrase.

Now this gives a very nice idea about the structure of the sentence, in this sentence, I am having a noun phrase and a verb phrase, verb phrase contains a verb and noun phrase and noun phrase; you can either put a single word like Joe or you can put multiple words like the man from Amherst similarly in that noun phrase you can put potatoes or it becomes noun phrase like beautiful russet potatoes. So, this gives me lots of idea about how words are grouped and arranged together.

(Refer Slide Time: 09:20)

*Evidence that constituency exists*

*They appear in similar environments*

*Kermit the frog comes on stage*  
*They come to Massachusetts every summer*  
*December twenty-sixth comes after Christmas*  
*The reason he is running for president comes out only now.*  
But not each individual word in the constituent  
\*The comes our... \*is comes out... \*for comes out...

*Can be placed in a number of different locations*

Constituent = Prepositional phrase: *On December twenty-sixth*  
*On December twenty-sixth I'd like to fly to Florida.*  
*I'd like to fly on December twenty-sixth to Florida.*  
*I'd like to fly to Florida on December twenty-sixth.*  
But not split apart  
\*On *December* *I'd like to fly twenty-sixth to Florida.*  
\*On *I'd like to fly December twenty-sixth to Florida.*

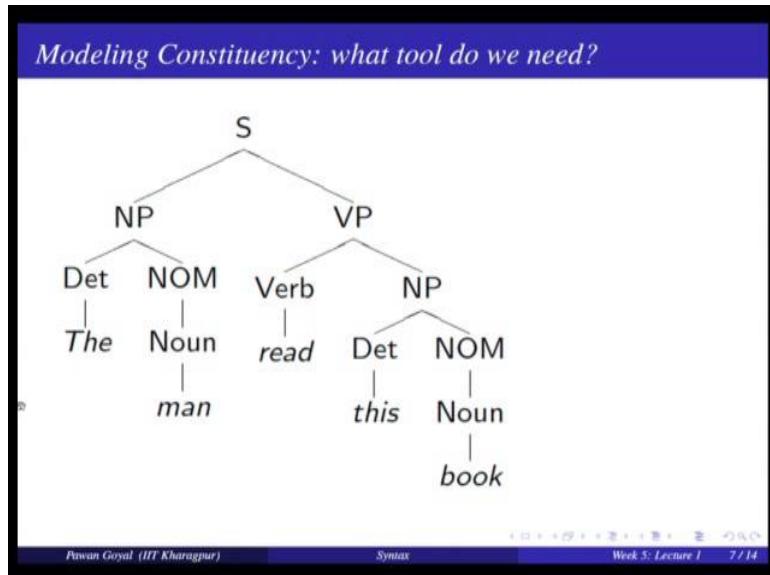
Ptwan Goyal (IIT Kharagpur) Syntax Week 5: Lec

Now, there is some evidence that questions actually exist in language, yes, there are 2 different evidences, one is that this phrase appears in very very similar environment. So, far I have talk about the 4 phrases that noun phrases that discussed in one of the various phrases like let we see this examples Kermit, the frog comes on stage, note it, they come to Massachusetts every summer, December 26th comes after Christmas, the reason he is running for president comes out only now. So, all these 4 nouns phrases are coming in a very similar context of the word say comes yes, but I cannot take any individual word from here and put that in the context. So, I cannot take the word the, and say the comes out or I cannot say is comes out and I cannot say for comes out in this sentence, I can only say the reason he is running for president comes out, only now not an individual word. So, this whole thing behaves as a single unit and they occur in similar context all the noun phrase occurring in similar context, this is got evidence that is got the constituency, each has to be there.

What is the other evidence? So, the evidence is that this whole phrase is together can locate many locations in the same center. So, for example, if I take on December 26 that is the prepositional phrase and I can put it in many different phrases in the sentence on December 26, I would like to fly to Florida, I would like to fly on December 26 to Florida or I would like to fly to Florida on December 26, all 3 are valid sentences where this complete phrase on December 26 as we put in multiple location, but you cannot break this into 2 parts and put it in phrases. So, you cannot say on December I would like to fly 26th to Florida or on I would like to fly December 26th to Florida you cannot say that. So, this will as a single group it

cannot be split off part. So, these are some evidence that constituency actually exists in the language.

(Refer Slide Time: 11:44)



Now, what is the formal tool by which we can model this constituency? That is how words are arranged together which words come together which was do not come together and what groups make a sentence and what groups make a verb phrase what group make a noun phrase; how can I model all that what is the formulation and if you have take any course on formal language in automated theory all theory computation you might already know that the formulate that we can use that in context free grammar.

(Refer Slide Time: 12:21)

**Modeling Constituency**

**Context-free grammar**  
The most common way of modeling constituency

**Consists of production Rules**  
These rules express the ways in which the symbols of the language can be grouped and ordered together

**Example**  
Noun phrase can be composed of either a ProperNoun or a determiner (Det) followed by a Nominal; a Nominal can be more than one nouns  
 $NP \rightarrow Det\ Nominal$   
 $NP \rightarrow ProperNoun$   
 $Nominal \rightarrow Noun\ | Noun\ Nominal$

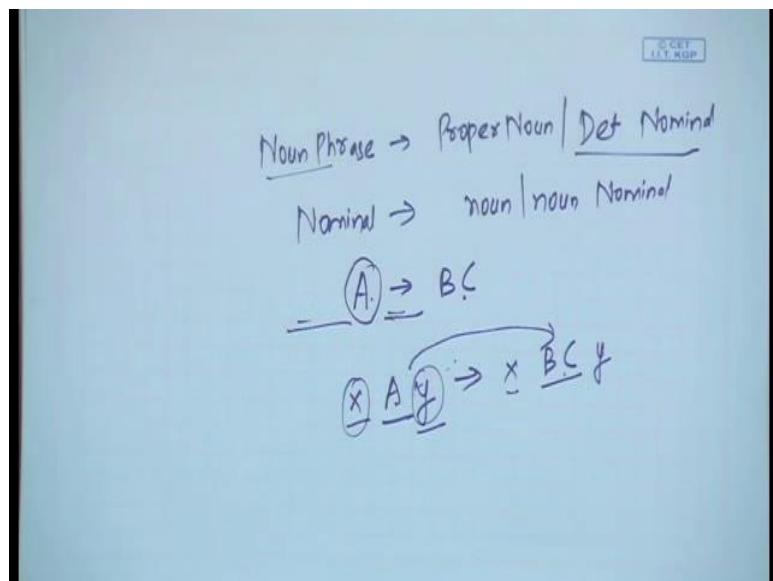
Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 1      8 / 14

This is the most common way of modeling constituency. So, in one of the earlier lectures we talked about the (Refer Time: 12:28) languages by using deterministic finite automata of finite automata. So, this is context for context free languages by using by the context free grammar.

Now, so I will not going in very very basics, I will just talk briefly how do we use context free grammar, I will also defined the notions for the context free grammar. So, in the case of context free grammar what you will have you will have some sort of production rules that is what we mostly interested in. So, what they do the production rules will try to express what are the way in which various symbols of the language of group together and that is our main interest in using context free grammar which symbols are being group together that I can use that I can find out using or I can express using the production rules in context free grammars.

Let us see one simple example. So, I want to model this (Refer Time: 13:27) language that is noun phrase can be composed to either a proper noun or a determiner followed by a nominal where a nominal can be more than one nouns that is something I want to express about language. So, how do I use the context free grammar? So, I will say noun phrases proper noun or determiner nominal the nominal is noun or many nouns.

(Refer Slide Time: 14:03)



How I write in context free grammar, I will say noun phrase is proper noun or determiner nominal and what is a nominal? It is one noun or more than one noun. So, noun or noun by a nominal this is the regression here. So, I can allow as many numbers of nouns as I want by using this and that is my context free grammar for denoting the symbol that is the idea I can express all these facts about language how words are groups together which groups come together by using this production rules.

Once we know that what is the formulation of context free grammar very briefly said so, in context free grammar when we study when we talk about a quadruple so, therefore, 4 important variable set of variables.

(Refer Slide Time: 15:09)

The slide has a blue header bar with the title 'CFG for Languages'. The main content area is divided into two sections: 'CFG:  $G = (T, N, S, R)$ ' and 'Terminals and pre-terminals'. The first section contains a bulleted list of components: 

- $T$ : set of terminals
- $N$ : set of non-terminals
  - For NLP, we distinguish out a set  $P \subset N$  of pre-terminals, which always rewrite as terminals
- $S$  : start symbol
- $R$ : Rules/productions of the form  $X \rightarrow \gamma$ ,  $X \in N$  and  $\gamma \in (T \cup N)^*$

 The second section, 'Terminals and pre-terminals', states: 'Terminals mainly correspond to words in the language while pre-terminals mainly correspond to POS categories'. At the bottom, there is footer information: 'Ptwan Goyal (IIT Kharagpur)', 'Syntax', 'Week 5: Lecture 1', and '9 / 14'.

Firstly, I have set of terminals we have the leaf notes in my tree whenever we see. So, they will always come at the end and whenever I get a terminal I cannot derive anything from there. So, I have set of terminals. So, we will see what in the case of language what do they mean then we have set of non terminals that help me to do the derivations, these are the variable from which you can further derive in more (Refer Time: 15:36)

Now, so what is different in the case of NLP, what is some distinguish, we will make in the set of non terminals, we will also distinguish the set P that are pre terminals. So, pre terminals are those non terminals that will always derive terminals. So, they will always give me leaf nodes and with the example that will be clear what are what are they in the case of language.

Then I have start symbol from which I am starting my derivation. So, if I have to model the sentence I should be starting with S that is the sentence and then I have the rules and they always of form X going to gamma and X has to be a non terminal, a single non terminal and gamma can be any sequence of terminals and non terminals and that is the constraint that we see in case of context free grammar. So, this is the quadruple and we also seeing a pre terminal that is the set of non terminals in the language, what are terminals and pre terminals? Terminals in the language will mainly with the final words that I will see in the lexicon and pre terminals will be part of speech categories from the pre because from the pre terminals you can derive only terminals.

(Refer Slide Time: 17:02)

The slide has a blue header bar with the text 'CFG for Languages'. Below it is a green box containing the word 'Example'. Inside the green box, there are several grammar rules:  
 $NP \rightarrow \text{Det Nominal}$   
 $NP \rightarrow \text{ProperNoun}$   
 $\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$   
Now, these can be combined with other rules, that express facts about a lexicon.  
 $\text{Det} \rightarrow \text{a}$   
 $\text{Det} \rightarrow \text{the}$   
 $\text{Noun} \rightarrow \text{flight}$

Below the green box is a pink box containing the question: 'Can you identify the terminal, non-terminals and preterminals?' To the right of this box is a circular portrait of a man with glasses and a blue shirt.

At the bottom of the slide, there is a navigation bar with icons for back, forward, and search. The text 'Ptwan Goyal (IIT Kharagpur)' is on the left, 'Syntax' is in the center, and 'Week 5: Lexicon' is on the right.

Let us see one example and then we can point out what are terminals pre terminals and non terminals. So, this is what we are modeling earlier what is the noun phrase determiner for the nominal or a proper noun and where a nominal is a noun or a set of nouns which I model using a noun followed by a nominal now are you see some terminals here. So, there are no words. So, there is no terminal. So, I cannot use that to derive a phrase or a sentence it can only give me a set of grammatical categories. So, I have to include some facts on the lexicon to make it a complete grammar.

For example, I can include some determiners some nouns and some proper nouns. So, here I am including a and the has to determiners and flight as a noun now here can you identify what are the terminal non terminals and pre terminals. So, terminals are the words in my lexicon. So, a the and flight are my terminals pre terminals are the grammatical categories or parse category that will always give me terminals can you see that determiner noun are only giving me terminals. So, these are my pre terminals and apart from that all the variables like N P nominal they are my non terminals. So, proper noun no example is given, but proper noun is also a pre terminal it is part of speech category it will give you some words to the lexicon. So, these are my terminals non terminals and pre terminals.

(Refer Slide Time: 18:40)

*CFG as a generator*

$NP \rightarrow \text{Det Nominal}$   
 $NP \rightarrow \text{ProperNoun}$   
 $\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$   
 $\text{Det} \rightarrow \text{a}$   
 $\text{Det} \rightarrow \text{the}$   
 $\text{Noun} \rightarrow \text{flight}$   
Generating 'a flight':  
 $NP \rightarrow \text{Det Nominal}^*$   
 $\rightarrow \text{Det Noun} \rightarrow \text{a Noun} \rightarrow \text{a flight}$

- Thus a CFG can be used to randomly generate a series of strings
- This sequence of rule expansions is called a derivation of the string of words, usually represented as a tree

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 1

Now once you have the CFG, you can use that to generate various phrases or sentences in language. So, for example, I here this is the CFG for non phrases. So, can I can I generate a flight the phrase a flight is in this context free grammar. So, I will have to start with n p and I have to generate a flight. So, what is the first derivation I will to do from NP I will take this rule determiner followed by a nominal yes now determiner will give me a now from nominal I cannot go to flight directly. So, for nominal I will have to first get a noun and from noun I get the word flight. So, NP gives me determine nominal gives me noun determiner gives me a noun and a flight that is how it generate a sequence of words using this grammar and now you can do it for any sentence you can define a grammar for a sentence and generate a sentences from that.

What we are seeing here? So, there is a context free grammar is generating a series of strings and this sequence of rule expansions. So, the sequence that that you are using starting from NP going to determiner nominal then determiner noun then a noun then a flight this is called the derivation of the string using this grammar and we use it is tree structure to represent this derivation remember one of the very first tree that we had shown as a motivation. So, we will try to come up with such trees using these derivations and when we will be call the past tense.

(Refer Slide Time: 21:20)

*CFGs and Grammaticality*

A CFG defines a formal language = set of all sentences (string of words) that can be derived by the grammar

- Sentences in this set are said to be **grammatical**
- Sentences outside this set are said to be **ungrammatical**

Ptwan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 1

Now, what is the notion of grammaticality using context free grammars the idea is that you defined one grammar for your language you have to assume that this is the only grammar. So, any tool that is not expressed in the in the grammar is not allowed in the language. So, now, when you are given a new sentence if you can see that there is a way to generate this sentence using my grammar the sentence is grammatical as per my grammar if the sentence cannot be generated in the grammar this is not grammatical this is a simple notion or using this grammar whatever sentence I can generate is grammatical and whatever I cannot is not grammatical. So, it depends on the grammar that you have designed the context free grammar that you have designed so yes. So, whatever can be derived is grammatical and others are ungrammatical.

(Refer Slide Time: 21:27)

*CFGs and Recursion*

*Recursive Definition*

- $PP \rightarrow \text{Prep NP}$
- $NP \rightarrow \text{Noun PP}$

*Example Sentence*

[<sub>s</sub>The mailman ate his [<sub>NP</sub> lunch [<sub>PP</sub> with his friend [<sub>PP</sub> from the cleaning staff [<sub>PP</sub> of the building [<sub>PP</sub> at the intersection [<sub>PP</sub> on the north end [<sub>PP</sub> of town]]]]]]]]].

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 1

Now, CFGs are interesting because they can also model some very interesting phenomena in language syntax like recursion. So, in language you make lot of big sentences by doing recursion. So, for example, a preposition phrase can be written as a preposition followed by a noun phrase and noun phrase can be written as a noun phrase followed by a preposition phrase. So, you see there is a recursion here yes. So, I can encode the noun phrase noun and prepositional phrase is here preposition phase can again encode a noun phrase which can encode a proportion phrase. So, this is the recursion is very very common in language.

Let us see 1 example. So, the set example is the mailman ate his and this is the complete noun phrase is starting from lunch till the end of the sentence, lunch and noun phrase is a noun followed by a preposition phrase with and this is not shown here, but this is again starts a noun phrase preposition phrase is a preposition followed by a noun phrase with his friend from the cleaning staff and all this is a noun phrase and what is this noun phrase noun his friend followed by a preposition phrase and so on the recursion is very nicely captured by using context free grammar.

Now, shall end this lecture by just saying, what is the context denotes in context free grammar? What is the meaning of context? So, language we talk about context as such. So, we say context is given a word find out what is the context what are the previous words and what is the topic and all that. So, these are all define the context, but this context is nothing to do with what is context in the case of context free grammar this is very very formal notion.

(Refer Slide Time: 23:35)

The slide has a blue header bar with the text "What does Context stand for in CFG?". Below the header, there is a list of bullet points:

- The notion of *context* has nothing to do with the ordinary meaning of word context in language
- All it really means is that the non-terminal on the left-hand side of a rule is out there all by itself (free of context)

Below the list, there is a diagram illustrating a grammar rule  $A \rightarrow BC$ . It shows two bullet points:

- I can rewrite  $A$  as  $B$  followed by  $C$  regardless of the context in which  $A$  is found
- Or when I see a  $B$  followed by  $a:C$ , I can infer an  $A$  regardless of the surrounding context

At the bottom of the slide, there is a footer bar with the text "Ptwan Goyal (IIT Kharagpur)" on the left, "Syntax" in the center, and "Week 5: Lect" on the right. There is also a small circular portrait of the speaker.

This is nothing to do with the ordinary meaning of word context in language. So, all this means is that in context free grammar whenever I am doing a derivation. So, whenever I am writing A gives the B C or say noun phrase, give me determiner nominal whenever I am writing a rule like that it means is that the noun terminal left it all in its own by itself it does not need any context around it. So, I can always write a goes to b c irrespective of whatever is around my word a. So, even if I have x a earlier I can use a to derive x b c and if I have irrespective of what is x or any y I can always write like this and this x and y are immaterial they do not matter it can be null it can be whatever.

Similarly, if I inferring whenever I see A B C, I can always infer A so, this; it might have come from A independent of the context of B C and this is what context free grammar. So, if you go would go the next label of context sensitive grammars there are you need the context this word a can derive b c in this particular left context in a particular right context we do not need this left and right context in the case of context free grammars this what the context the word context means in this case.

Whenever I have a rule, A goes B C, it means that I can write A, I can always write from A B followed by C, regardless of the context in which A is found or whenever I find and B followed by A C, I can infer a regardless of the context in which B and C is found. So, this is CFG for us, I am not going to lot of basics of context free grammars and I suggest that you can quickly look at any of the chapters in the basic books of formal languages and automated

theory so, but whatever is required for our tasks of doing parsing, I have covered in this lecture and I will cover the necessary things in the next lecture.

In the next lecture, what we will do? We will start from CFGs and we will see how we can use that for actually doing the parsing for a given sentence in the language. So, we will take a various approaches for parsing so that will be the next topic for the next lecture.

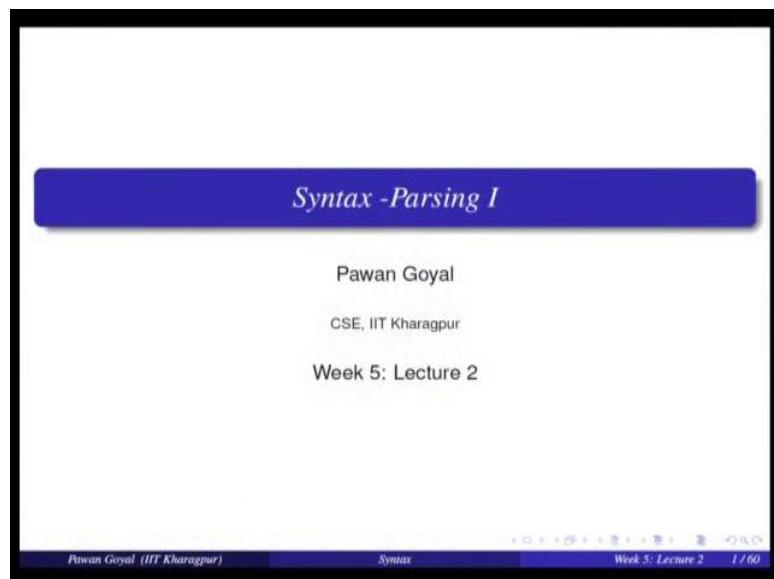
Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 23**  
**Syntax - Parsing I**

So, welcome back for second lecture of this week. So we had started our discussions on Syntax.

(Refer Slide Time: 00:24)



So in the last lecture we had talked about what is the formulation for doing the syntax parsing. So we had talked about context free grammars. Now we will see how do we use context free grammars for the actual parsing.

(Refer Slide Time: 00:46)

**Grammar Rewrite Rules**

$S \rightarrow NP\ VP$	Det $\rightarrow$ <i>that   this   a   the</i>
$S \rightarrow Aux\ NP\ VP$	Noun $\rightarrow$ <i>book   flight   meal   man</i>
$S \rightarrow VP$	Verb $\rightarrow$ <i>book   include   read</i>
$NP \rightarrow Det\ NOM$	Aux $\rightarrow$ <i>does</i>
$NOM \rightarrow Noun$	
$NOM \rightarrow Noun\ NOM$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	

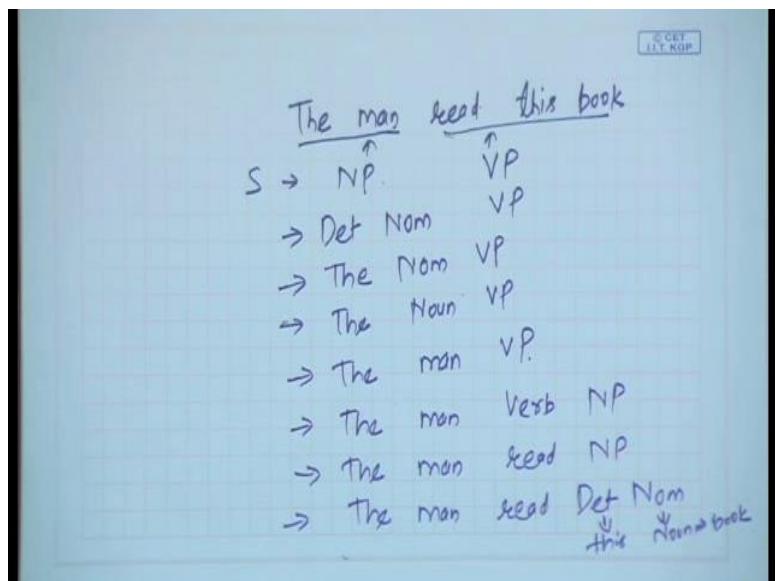
$S \rightarrow NP\ VP$

- $\rightarrow$  Det NOM VP
- $\rightarrow$  *The* NOM VP
- $\rightarrow$  *The Noun* VP
- $\rightarrow$  *The man* VP
- $\rightarrow$  *The man* Verb NP
- $\rightarrow$  *The man* read NP
- $\rightarrow$  *The man* read Det NOM
- $\rightarrow$  *The man* read *this* NOM
- $\rightarrow$  *The man* read *this* Noun
- $\rightarrow$  *The man* read *this book*

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lec 1

So let us take a simple grammar in terms of in a CFG formulation. So you have in the grammar it is said that a sentence can be a noun phrase followed by a verb phrase or an auxiliary followed by a noun phrase followed by verb phrase or single verb phrase. Noun phrase can be a determiner followed by a nominal can be a noun and so on. So all the various possibilities are shown here and similarly these are all the pre terminals that are giving me terminals that are words in my lexical. So this is one such grammar. And now using this grammar I can write various sentences. So for example, I want to write the sentence - the man read this book. So I want to write this sentence the man read this book. Now how do I what kind of rules in the grammar do I use so that I can generate the sentence. So I have to write the man read this book.

(Refer Slide Time: 02:10)



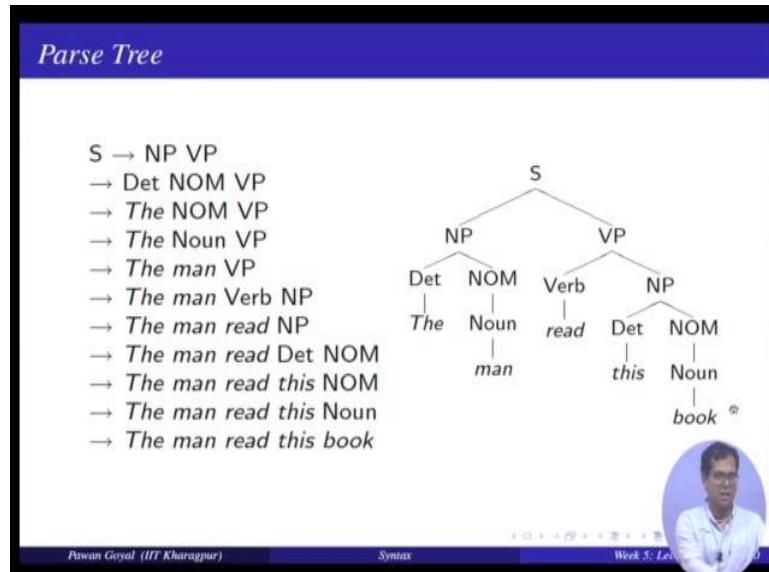
So let me just do it quickly on the paper. Now the sentence can be a noun phrase followed by a verb phrase or an auxiliary followed by a noun phrase verb phrase or a verb phrase. In this case it does not start with an auxiliary nor a verb. So it has to be a noun phrase followed by a verb phrase. So I have to first start by a noun phrase followed by a verb phrase. And I will assume that the man will come in noun phrase and read this book come in verb phrase.

So what will be the next derivation? So NP have to get the man. So I have to get something that says determiner nominal. So here I will say determiner nominal and verb phrase. And determiner can give me a nominal can give me noun which can give the word man. So in some steps I can determine to the nominal verb phrase the noun verb phrase. And if I want to complete this I will say the man verb phrase. Now the verb phrase I have to derive read this book. So verb phrase goes to verb and noun phrase is the something like that yes verb phrase goes to verb and noun phrase.

So I will use this. And say the man verb noun phrase verb can directly give me read, the man read noun phrase. And from noun phrase I have to derive this book. So I have to again write this determiner for a nominal. This will become a determiner the man read determiner nominal. And this will give me this, and this will give me noun. And this will give me book. And that will give me the whole sentence the man read this book. That is, I derive a sentence using this grammar.

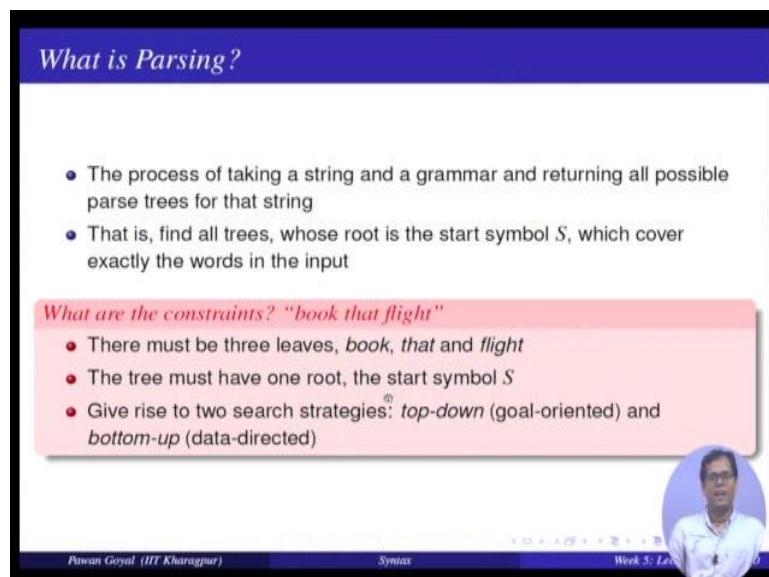
So now once I have this derivation I can use this derivation to denote what is the particular syntax tree that generalizes the sentence as per this grammar.

(Refer Slide Time: 04:40)



So same derivation can be used for to show the parse tree, so this is my derivation and this derivation can be shown using this parse tree. In the question of sentence, you derived NP VP from NP you derived determiner and nominal VP verb and noun phrase. Noun phrase to this particular sub, so this is exactly what was the derivation that you did for the sentence in my grammar.

(Refer Slide Time: 05:09)



So that can immediately tell me what is the problem of parsing. I am given the grammar and I am given the sentence in my language. So given the sentence in my grammar I want to find out what is it is actual parse tree. And they may not be unique parse they may be multiple parse tree. So the process of parsing is to find out all the possible parse trees for a given sentence as per my grammar. So find out all the trees whose root is the start symbol S because I have to start with the start variable in my context free grammar and which cover exactly the words in the input.

Now, what are the constraints on which I do the parsing? Suppose my sentence is book that flight. So what are the constraint? In my parse they should be 3 leaf nodes. Book that and flight and MS is you start at the start node. And then I have to explore all the possible rules such that I come up with the tree is starting with us as the root node and book that flight as the only 3 nodes 3 leaf nodes. So yes the tree must have one root the start symbol S and that tells me that I can explore it in at least 2 different ways. One is I can with top down you start from S try to find out all the possible trees in finding out I can have as per my grammar and see if there is one tree that can give me the only the leaf nodes book that flight this is top down.

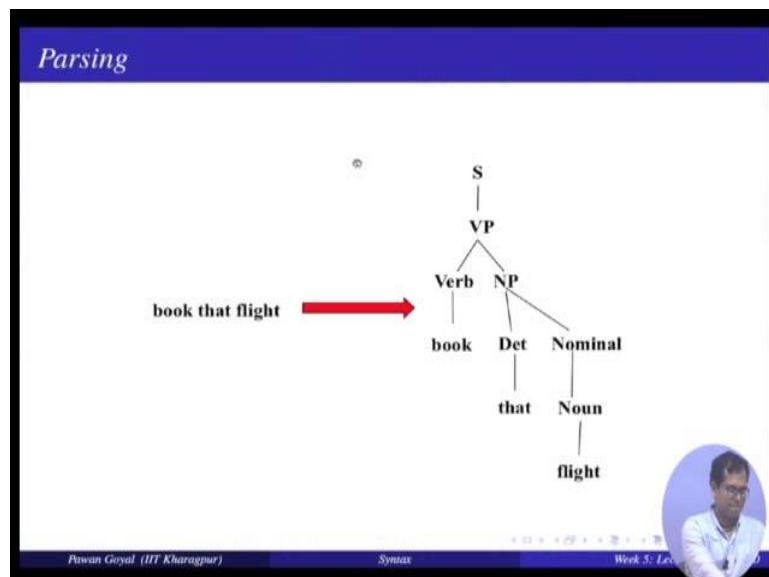
Other approach is bottom up. I start from book that flight go up words what are the non-terminals and then see if any of this combination can lead to a full tree it starting from S. So these are 2 different strategies. And that is what we will discuss how do we use it top down or bottom up approach for parsing given this grammar.

(Refer Slide Time: 07:09)

Parsing	
Grammar	Lexicon
$S \rightarrow NP\ VP$	$Det \rightarrow the   a   that   this$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book   flight   meal   money$
$S \rightarrow VP$	$Verb \rightarrow book   include   prefer$
$NP \rightarrow Pronoun$	$Pronoun \rightarrow I   he   she   me$
$NP \rightarrow Proper-Noun$	$Proper-Noun \rightarrow Houston   NWA$
$NP \rightarrow Det\ Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	$Prep \rightarrow from   to   on   near   through$
$Nominal \rightarrow Nominal\ Noun$	
$Nominal \rightarrow Nominal\ PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	
$VP \rightarrow VP\ PP$	
$PP \rightarrow Prep\ NP$	

So let us take this grammar. So you have certain rules. So left hand side is mainly the rules for non-terminals and pre terminals and right hand side are for the lexicon, pre terminals deriving the terminals. Now using this grammar, you want to find out the parse for book that flight. And we will use both the top down and bottom up strategies for finding out this parse tree.

(Refer Slide Time: 07:40)



So yeah this is also what is the expected parse tree. Book that flight this is your whole verb phrase, it is starting from a verb and a noun phrase. So a book is a verb and this that

flight is a noun phrase. So I want to come up with this parse tree given the grammar, and how do I do that in a deterministic manner using the using my top down strategy

(Refer Slide Time: 08:19)

### Top-Down Parsing

- Searches for a parse tree by trying to build upon the root node  $S$  down to the leaves
- Start by assuming that the input can be derived by the designated start symbol  $S$
- Find all trees that can start with  $S$ , by looking at the grammar rules with  $S$  on the left-hand side
- Trees are grown downward until they eventually reach the POS categories at the bottom
- Trees whose leaves fail to match the words in the input can be rejected

Pawan Goyal (IIT Kharagpur) Syntax Week 5: Lecture 2 7 / 60

So how do I start? So I have to start from my start node root node in top down and using my grammar as my grammar I will see what is the different possible rules I can apply at this point, keep on going downwards.

(Refer Slide Time: 09:20)

### Top-Down Parsing

```
graph TD; S --- NP; S --- VP; NP --- Pronoun; Pronoun --- book;
```

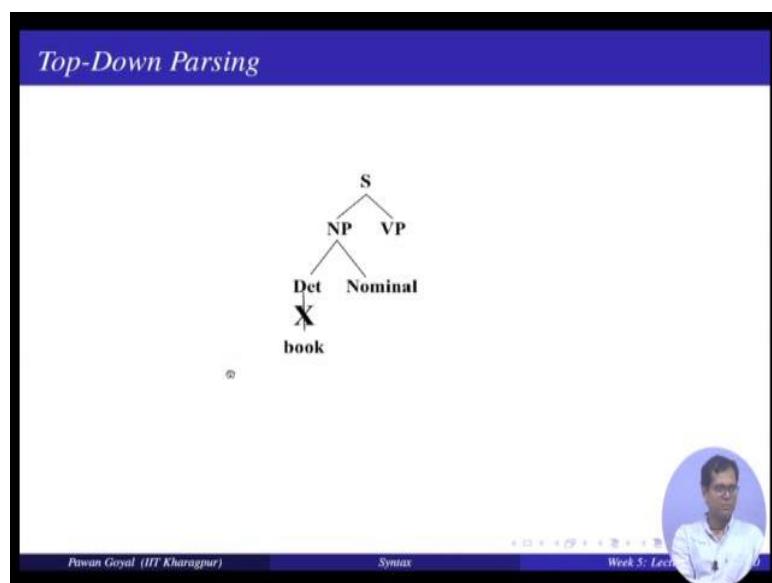
Pawan Goyal (IIT Kharagpur) Syntax Week 5: Lecture 2 7 / 60

So a start by assuming that the input can be derived by S, then find out the trees that is start with S looking at the rules that are having on the left hand side because what are the all are different things that S can derive. Now when you are going downward in your tree, once you obtain the part of speech category you will see if that matches the word in the leaf nodes. If it is not matching you will go back and try out some other path. And if there are any trees where the part of speech categories is not matching the words at the leaf nodes you will get (Refer Time: 09:19). So let us see. So I am starting with S.

Now what is my first rule? So what are the different rules from S.? So first rule is S can go to NP VP. So what I will do? I will try to explore that path. Scan got to NP VP. Now from NP what is the next possible rule? So next rule from NP is NP can give me a pronoun, NP give me a pronoun. Again I will try to explore this further. The pronoun gives me the pronoun will be the first word. Now pronoun is the pre-terminal.

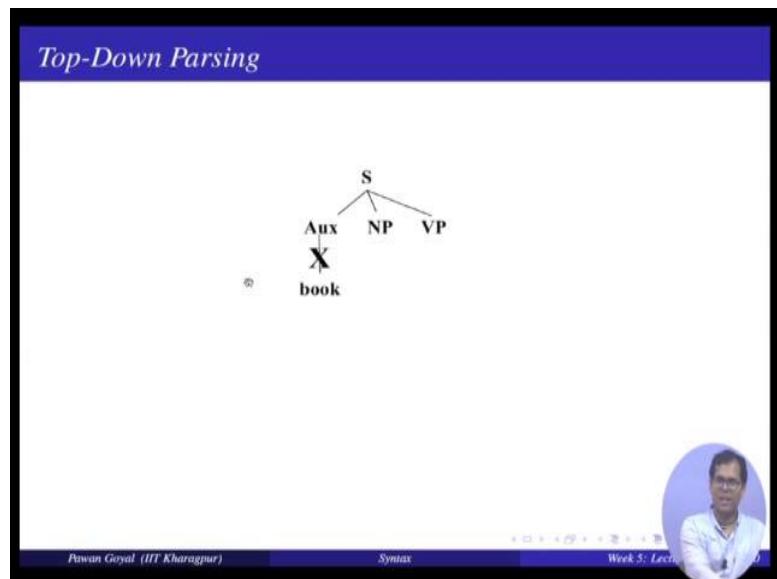
Now what is the first pre terminal book? So pronoun can never give me a book. Book is not a pronoun. So this part is not correct. So I will go back from NP I will try to expose some other path. So NP can also give me a proper noun. So you see these are the rules in sequence in my grammar, a NP can give me a pronoun NP can give me proper noun. So again proper noun is a pre terminal that cannot give me a book. So again I cannot accept this path. So I will go back. Next rule is NP gives me a determiner followed by a nominal.

(Refer Slide Time: 10:36)



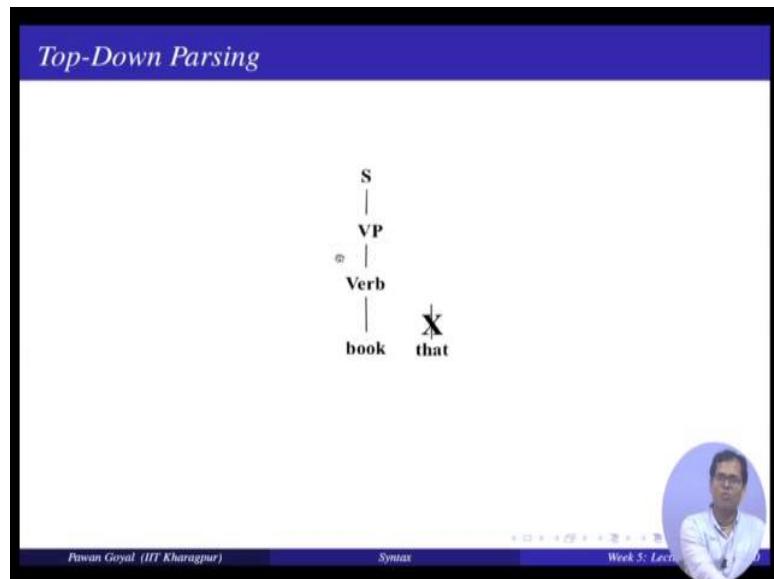
So again the first word has to be determiner that in the sentence, but the first word is book that is not a determiner. So again this path has to be this path is not correct. Now we see in the grammar there is no other rule that has NP on the left hand side so; that means, I have to now go back to the initial assumption that S will derive NP VP. So I have to try out the next possibility, as per my grammar the next possibility is S can give me auxiliary followed by noun phrase followed by a verb phrase. So let me do that.

(Refer Slide Time: 11:18)



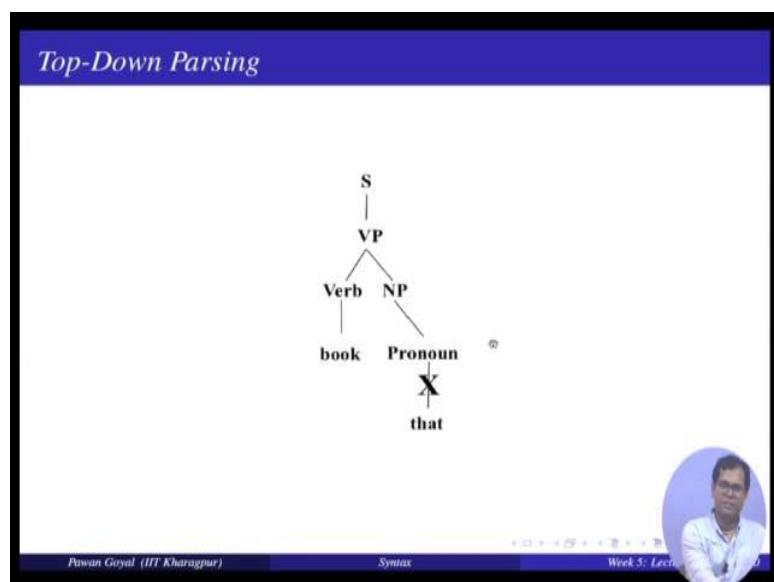
So S give me an auxiliary noun phrase and verb phrase. Again what is auxiliary? Auxiliary is a pre terminal that gives only does it does not give me book. So again this part is not correct and auxiliary gives me anything else. So again I go back and try out something else from S and the only remaining thing is VP.

(Refer Slide Time: 11:39)



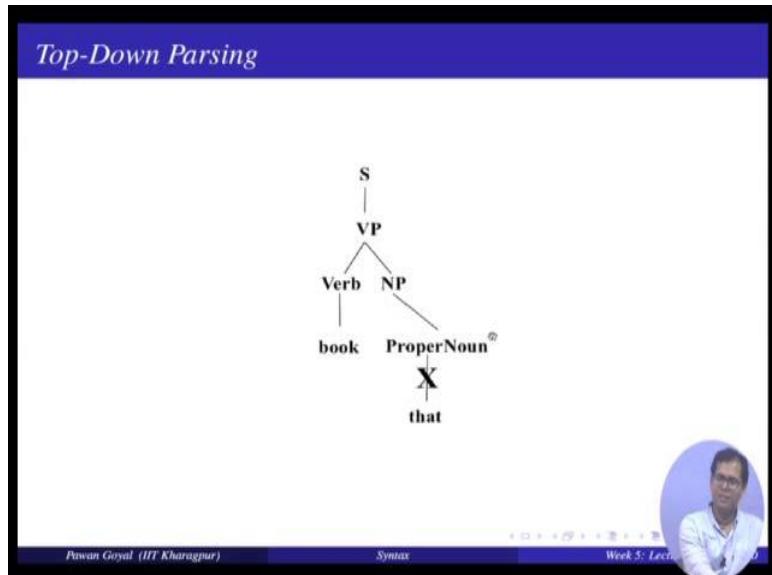
So S gives me a VP. Now I will go to my grammar. What are the rules from VP? VP can give me a verb that is the first thing, VP gives me a verb good. Now verb also gives me book. That that matches there, but what happens to the other 2 words in my sentence that flight. Verb gives me book, but that flight is not covered in this tree. So this is not a valid tree, yes that flight there is no other node starting from S that captures this this is again not a valid tree. So I will try out something else with VP. So next possibility is verb followed by NP.

(Refer Slide Time: 12:22)



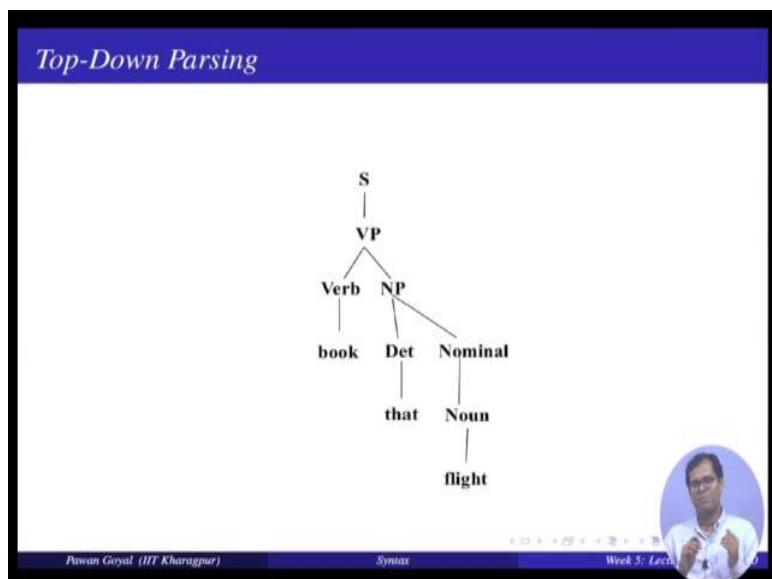
Verb followed by NP. And verb gives me book. Well now from NP I have to get that flight. Now again from NP I can get pronoun.

(Refer Slide Time: 12:41)



Pronoun cannot give me that yes. Then I can get proper noun. It will also not give me that. Then I can from NP I can get determiner followed by a nominal.

(Refer Slide Time: 12:47)



And that is a determiner and from nominal I can go to noun and this can give me flight. So that means, by doing all these explorations systematically I can come up with a parse

tree that starts from S and exactly covers these leaf nodes book that flight in my input. So that is my top down parsing strategies. So hope that is clear.

So we are given all the rules it starts from S and try to explore in some order. You can take it you can try to explore in the same order that is in which they are given to you. You might try to put them in the order in which they are actually used in language. So which one is more probable than other that is also. But again you take can you see that this requires a far too many steps right. So you are exploring paths that will probably never lead to the whole parse trees. So this may need very certain visual space. So we will try to take that problem that how we can avoid that. So this is my a, this was my top down parsing.

(Refer Slide Time: 13:59)

The slide has a blue header bar with the text "Bottom-Up Parsing". The main content area contains two bullet points:

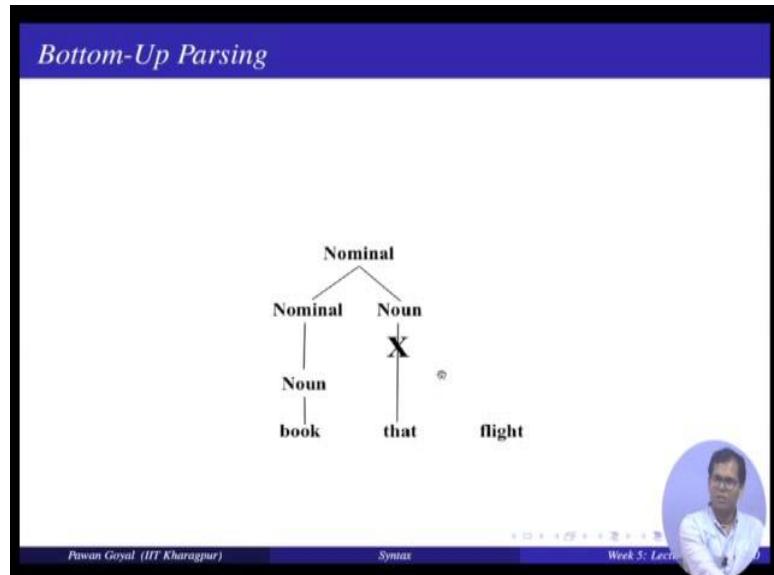
- The parser starts with the words of the input, and tries to build trees from the words up, by applying rules from the grammar one at a time
- Parser looks for the places in the parse-in-progress where the right-hand-side of some rule might fit.

At the bottom of the slide, there is a video player interface. It shows a circular thumbnail of a person, the name "Pawan Goyal (IIT Kharagpur)" next to it, and the word "Syntax". To the right of the thumbnail, there is a small video frame showing a person speaking. Below the video frame, the text "Week 5: Lec 10" is visible. The video player has standard controls for play, pause, and volume.

Now what do I do in bottom up parsing. In top up I start with S. In bottom up I will start with my leaf nodes. I will start with book that flight. And I will try to go my tree upwards and see which one can give me if complete tree it is starting from S. So the parser starts with the words of the input and tries to build trees from the words up by applying rules from the grammar one at a time. And parser looks for the places in the parse in progress where the right hand side of the rule might fit. In top down we were looking at always the left hand side, if the current non-terminal what is the rule in the left hand side. So that accordingly I will I am trying to generate the right hand side. Here I

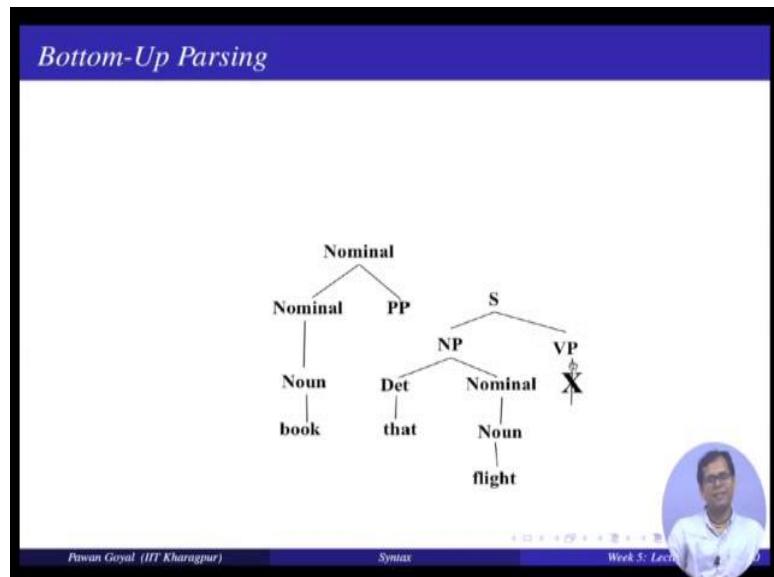
am seeing what in the right hand side I already have accordingly I select a rule in my grammar.

(Refer Slide Time: 14:52)



So let us do this bottom up parsing. So I have the sentence book that flight. I will start by seeking what are the nodes in my grammar that can that can generate this. So I start by say book. And the first rule that their word book is noun. Then I go up to noun and say nominal gives me noun, fine. And nominal can gives me nominal followed by a noun. So I am going my tree upward, but now I arrive at noun that is a pre terminal and noun will not give me that. So that is not a noun. So this is the inconsistency. So I have to go back and see.

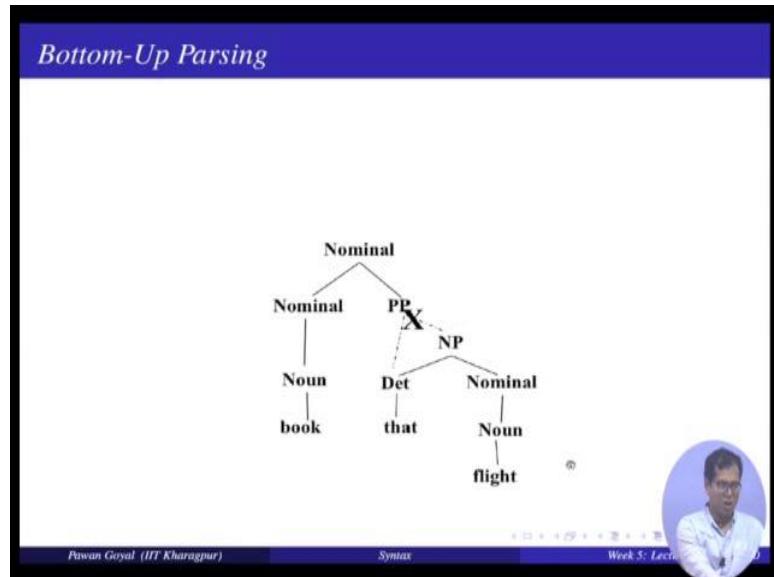
(Refer Slide Time: 15:38)



What is the next possible rule from nominal? So I have to derive from nominal followed by PP. So remember, we are seeing in which rule this occurs in the right hand side and accordingly I will take the production. Nominal gives me nominal followed by PP, now can PP give me that flight? So from PP, so from that I try to go it upwards I see determiner. Determiner comes in right hand side of this rule NP gives me determiner nominal. And flight again I grow it upwards it gives me noun a nominal can give me noun. So this looks a nice tree.

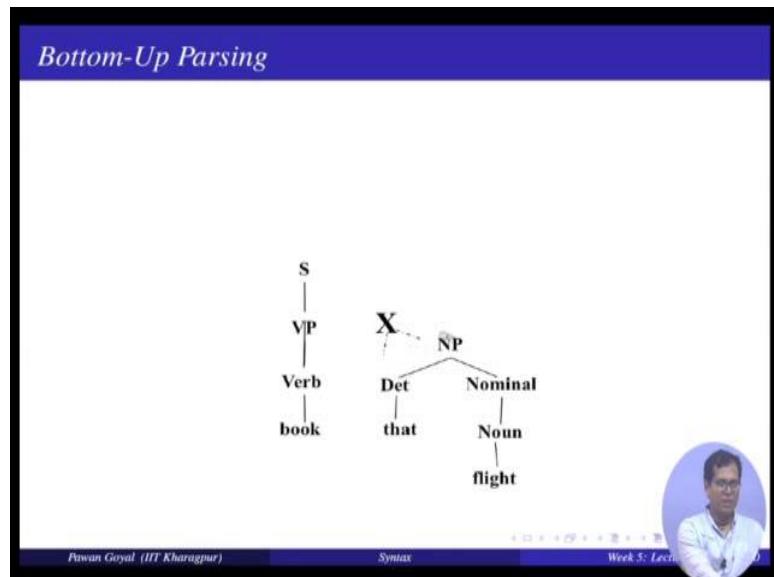
But now can I attach PP to NP. So S if I go NP upwards I will say S can give me NP followed by VP. This creates a problem in that this verb phrase does not have anything in. So it cannot take me to any leaf node.

(Refer Slide Time: 16:46)



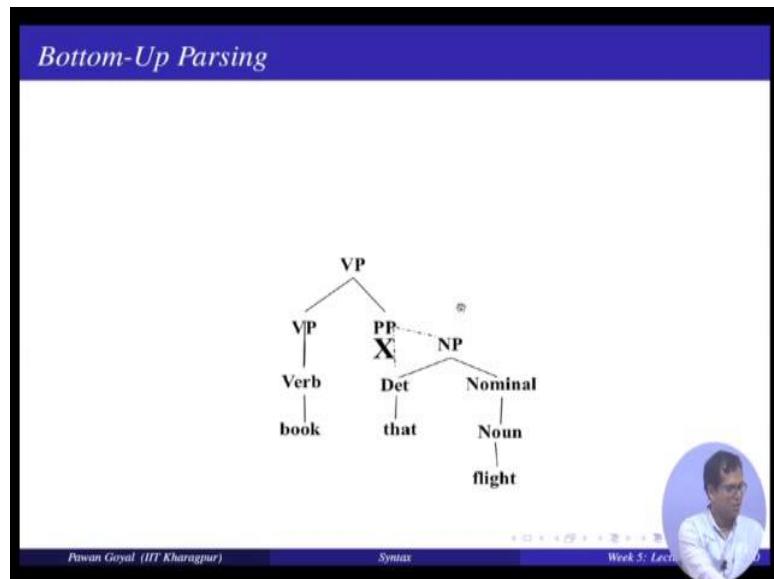
So this is not valid. If I go to this part can PP attached to n determiner and NP again there is no rule in my grammar that PP can be determiner followed by NP.

(Refer Slide Time: 16:58)



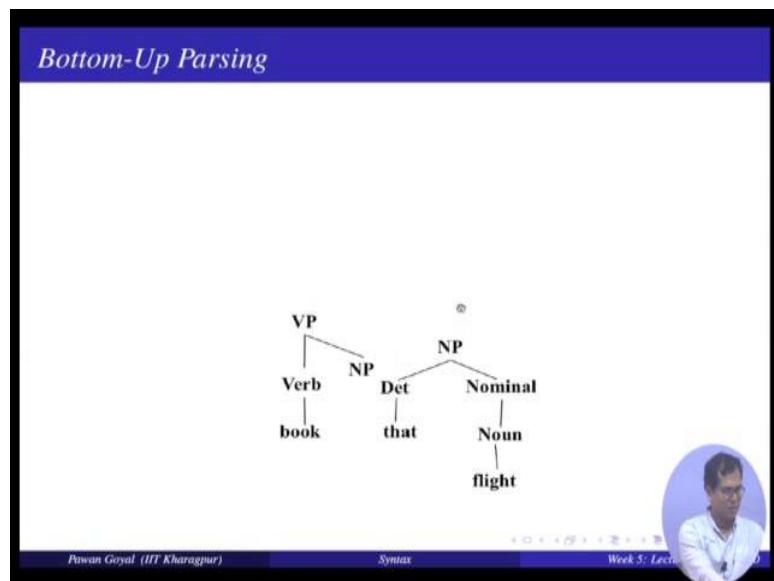
So I have to go back again. Try some other paths we have got on this is right hand side. VP S, S and determiner nominal and noun phrase, what can they give to me S again this does not work out.

(Refer Slide Time: 17:13)



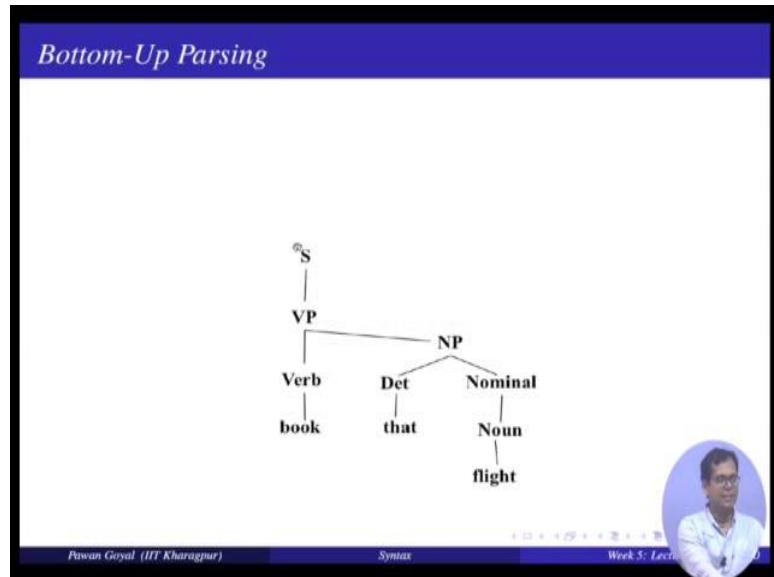
So I try out something else with VP. So I say VP gives me VP followed by a prepositional phrase. Again prepositional phrase does not give me determiner followed by a noun phrase, this we will not work out. Again go one step down.

(Refer Slide Time: 17:28)



So earlier I was saying VP is deriving a verb, now I am saying VP derives verb followed by an NP does that work.

(Refer Slide Time: 17:40)



Yes, if I can attach this VP to this verb and this noun phrase. And then I can say that S can derive this VP and this gives me the whole parse tree. So this is my bottom up strategy.

I start with the words in the leaf in the leaves. Try to grow them upwards by seeing what is the rule in my grammar where this occurs in the right hand side. Similarly, I will see here and so on. And finally, can I build them as a single tree starting from S. And that strategy finally, gives me this particular tree. So if we just try to compose this top down versus bottom up approaches. So what do we see?

(Refer Slide Time: 18:25)

*Top-Down vs. Bottom-Up*

- Top down never explores options that will not lead to a full parse, but can explore many options that never connect to the actual sentence.
- Bottom up never explores options that do not connect to the actual sentence but can explore options that can never lead to a full parse.
- Relative amounts of wasted search depend on how much the grammar branches in each direction.

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 1

So what we were see in top down, we were always seeing wherever this non-terminal appears in the left hand side and I immediately do the production.

So what happens that I can  $x$ , I can always explore options that will never lead to the full parse. I can always find options that will only give me one word. So all the options were given me a parse is starting from  $S$  in in the top down strategy, but it might happened that some options are giving me a particular parse tree where all the words are not covered or some extra words are covered. So this actual correction the actual sentence is not taken care. So in the bottom up it is the other way around. So we have always seeing the words first you are exploring that those paths that are covering the whole sentence, but it might happen that the parse that you come up with is not a complete parse because it is not having a root at  $S$ . So they may not be a full parse. And there are ways to this specimen both top down and bottom up.

So you are seeing you are experiencing lot many parse that are probably not valuable and this depends on branching of my grammar in either direction.

(Refer Slide Time: 19:51)

The slide has a blue header bar with the title "Dynamic Programming Parsing". The main content area contains a bulleted list:

- To avoid extensive repeated work, must cache intermediate results, i.e. completed phrases.
- Caching (memoizing) critical to obtaining a polynomial time parsing (recognition) algorithm for CFGs.
- Dynamic programming algorithms based on both top-down and bottom-up search can achieve  $O(n^3)$  recognition time where  $n$  is the length of the input string.

At the bottom of the slide, there is footer information: "Pawan Goyal (IIT Kharagpur)" on the left, "Syntax" in the center, and "Week 5: Lecture 2 57 / 60" on the right, along with standard presentation navigation icons.

So to avoid this problem and obtain an algorithm that works in polynomial time. So you will use some dynamic programming approach. So we have been using dynamic programming a lot in this course. So we use that for edit distance then we use that for (Refer Time: 20:07) coding kind of algorithm.

Now, we will see how to use that for obtaining an efficient parsing algorithm. So idea is that can we cache some intermediate results instead of exploring all the different possibilities that are not relevant. So by doing this caching I can obtain a polynomial time parsing algorithm for context free grammars. And there are different dynamic programming algorithms that are both top down as well as bottom up and they can work in roughly order of  $n^3$  time where  $n$  is the length of the sentence number of words in my input string.

So what are the different approaches for dynamic programming parsing?

(Refer Slide Time: 20:53)

The slide has a blue header bar with the title "Dynamic Programming Parsing Methods". The main content area contains a bulleted list:

- CKY (Cocke-Kasami-Younger) algorithm: bottom-up, requires normalizing the grammar
- Earley Parser - top-down, does not require normalizing grammar, more complex
- More generally, *chart parsers* retain completed phrases in a chart and can combine top-down and bottom-up searches.

At the bottom of the slide, there is footer information: "Pawan Goyal (IIT Kharagpur)" on the left, "Syntax" in the center, and "Week 5: Lecture 2 58 / 60" on the right.

So, one very popular algorithm is CKY algorithm that works in bottom up manner. So you will do it for individual words then you are going to sequence of 2 words and so on. Up to you go to the whole 6 sentence. And this is the only thing is that it requires some normalizing of the grammar that you will also see what is the normalization. Then there is early parser that is again very popular that works in top down manner. It does not require any normalization of grammar and slightly more complex than the CKY algorithm. And a generic frame work is something called a chart parser where for individual phrases in the sentence they will see what are the possible trees they were retain in the chart and use that for the higher level trees. So and they combined both of these approaches bottom up and top down.

So we will only focus on CKY algorithm that how do we use this for finding out parser in efficient polynomial time. So how does this CKY algorithm works? Now, before that we have seen that this requires normalization of grammar. So what is that?

(Refer Slide Time: 22:14)

The slide has a blue header bar with the text "CKY Algorithm". Below the header, there is a large white area containing two bulleted lists. The first list describes the conversion of a grammar to Chomsky Normal Form (CNF) with specific production constraints. The second list describes the bottom-up parsing process using a triangular chart. At the bottom of the slide, there is a footer bar with the text "Pawan Goyal (IIT Kharagpur)" on the left, "Syntax" in the center, and "Week 5: Lecture 1" on the right. A small circular video thumbnail of the speaker is also present in the bottom right corner.

- Grammar must be converted to Chomsky normal form (CNF) in which all productions must have
  - Either, exactly two non-terminals on the RHS
  - Or, 1 terminal symbol on the RHS
- Parse bottom-up storing phrases formed from all substrings in a triangular table (chart)

So to apply CKY algorithm, so my grammar must be converted to a normal form called Chomsky normal form. And what is the constraint in Chomsky normal form? So the constraint is that all the production of my grammar should be having one of these 2 forms. That is either exactly 2 non terminals on the right hand side or one terminal symbol on the right hand side.

(Refer Slide Time: 22:47)

The image shows handwritten notes on a light blue background. At the top, there is a diagram showing a production rule  $A \rightarrow \gamma$  with an arrow pointing to a circle labeled "CNF prod". Below this, another arrow points to a production rule  $A \rightarrow BC$  with the label "non-terminals". Further down, an arrow points to a production rule  $A \rightarrow a$  with the label "terminal". To the left, there is a partial diagram of a grammar with rules  $A \rightarrow B$  and  $B \rightarrow V$ . To the right, there is a partial diagram of a grammar with rules  $S \rightarrow X_1 VP$ ,  $X_1 \rightarrow NP VP$ , and  $S \rightarrow X_1 VP$ .

So what do I mean by that? In the context free grammars a rule is of the form  $A$  goes to  $\gamma$ , where  $A$  is a non-terminal and  $\gamma$  can be a sequence of terminals and non-

terminals. This is general CFC production rule. So what happens in the case of Chomsky normal form? The rules are constraint to be of these 2 forms A goes to B C. All these are non-terminals or A goes to small a small a is a terminal. So left hand side is always the same it is only one non-terminal. In Chomsky normal form what happens is that you put some constraints on the right hand side. So it can have either only 2 non terminals or only one terminal.

So to apply CKY algorithm, I must convert my grammar in any generic context free grammar form to Chomsky normal form. So what are the steps involved they are (Refer Time: 23:50) what are necessary for our case. So we will also see that how do we store all the possible phrases what are their parses in a triangular table in the CKY algorithm.

(Refer Slide Time: 24:10)

Converting to CNF	
Original Grammar	Chomsky Normal Form
$S \rightarrow NP VP$	$S \rightarrow NP VP$
$S \rightarrow Aux NP VP$	$XI \rightarrow Aux NP$
$S \rightarrow VP$	$S \rightarrow book   include   prefer$
$NP \rightarrow Pronoun$	$S \rightarrow Verb NP$
$NP \rightarrow Proper-Noun$	$S \rightarrow VP PP$
$NP \rightarrow Det Nominal$	$NP \rightarrow I   he   she   me$
$Nominal \rightarrow Noun$	$NP \rightarrow Houston   NWA$
$Nominal \rightarrow Nominal Noun$	$NP \rightarrow Det Nominal$
$Nominal \rightarrow Nominal PP$	$Nominal \rightarrow book   flight   meal   money$
$VP \rightarrow Verb$	$Nominal \rightarrow Nominal Noun$
$VP \rightarrow Verb NP$	$Nominal \rightarrow Nominal PP$
$VP \rightarrow VP PP$	$VP \rightarrow book   include   prefer$
$PP \rightarrow Prep NP$	$VP \rightarrow Verb NP$
$Pronoun \rightarrow I   he   she   me$	$VP \rightarrow VP PP$
$Noun \rightarrow book   flight   meal   money$	$PP \rightarrow Prep NP$
$Verb \rightarrow book   include   prefer$	$Pronoun \rightarrow I   he   she   me$
$Proper-Noun \rightarrow Houston   NWA$	$Noun \rightarrow book   flight   meal   money$

Pawan Goyal (IIT Kharagpur)

Syntax

Week 5: Lecture 2 60 / 60

So let me quickly see let me quickly show how we convert a grammar to Chomsky normal form. So in the left hand side we have a grammar, and I want to convert that the Chomsky normal form. So can you quickly see go through the grammar and find out the rules that are not in Chomsky normal form, so all these are one non-terminal giving me a one terminal. Yes, in all these cases. So they are pre terminal to terminal, they are always in Chomsky normal form, yes, a non-terminal giving me a single terminal. Now let me go upwards. So PP gives me preposition followed by a noun phrase, again one non-terminal giving me 2 non terminals in CNF. This is also in CNF, this is also in CNF, but this rule is not in CNF. VP goes to verb one non-terminal giving another non-terminal.

So how do I actually convert this to Chomsky normal form? So idea would be I will find out what are the terminals that this derives. So verb derives book include and prefer. So instead of this rule I will add a new rule, verb phrase gives me book include and prefer and this is the strategy. Similarly, here this is in CNF, this is in CNF. What about this rule nominal goes to noun? Again I will find out what are different things that noun derives book flight meal and money and I will add a rule nominal derives book flight meal and money this is in CNF.

So here again NP goes to pronoun I will find out pronoun goes to I he she me. So NP gives me I he she me. S goes to VP is again not in CNF. So VP gives me verb and also verb NP. So I have to add these rules S goes to verb NP S goes to VP NP S goes to preposition NP, plus for VP going to be verb at now takes it next rule VP verb going to book include prefer and the rule I will add is S goes to book include and prefer, S going to NP VP is fine.

But what do I do with this rule S goes to auxiliary NP and VP. So for this kind of rule S goes to auxiliary NP VP where a single non-terminal derives pre non-terminals. So what do I do? I coin some new non-terminals, so I will say that auxiliary noun phrase together makes new terminals say X1. So what will be my grammar it will be S goes to X1 followed by VP and X1 will be auxiliary followed by NP, and this is equivalent to this rule. And now this is in my Chomsky normal form.

So idea is that whenever you have a rule where you have 2 more than 2 non terminals you try to break them down such that it is one non-terminal another non-terminal this you can again further break down if needed. And this is a simple approach and if we have a rule like this A goes to B and B goes C you take this 2 a goes to small c this is a terminal. And also if B has something else you need take care of everything in this yes that is what we took we saw in this particular example there.

So now if I convert this (Refer Time: 27:59) grammar to Chomsky normal form, this is how it will look like. So you see that here S goes to VP now has multiple rules. S goes to verb NP S goes to VP NP and S goes to book include prefer and so on for the other case also.

So in this lecture we had seen that how do we do a simple parsing using top down approach and bottom up approach, but they are not very efficient, so can we do

something better by using dynamic programming parsing approach. And for that we are trying we will be seeing CKY algorithm. So how do you use CKY algorithm for an efficient parsing, but CKY algorithm requires normalization to some Chomsky normal form and we saw how do we convert a grammar to Chomsky normal form.

So in the next lecture we will start with this Chomsky normal form and see given a new, given a string how do we parse that using CKY algorithm.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture-24**  
**Syntax – CKY, PCFGS**

So, welcome back for the third lecture of this week. In the last lecture, we had discussed top-down and bottom-up parsing approach and then finally, we came up with the dynamic programming approach of using CKY algorithm; and we converted a grammar to Chomsky normal form. Now, once my grammar is converted to Chomsky normal form how does CKY algorithm work, so this is the idea.

(Refer Slide Time: 00:44)

The slide has a blue header bar with the title 'CKY Algorithm'. The main content area is white with a black border. It contains a bulleted list of steps for the CKY algorithm:

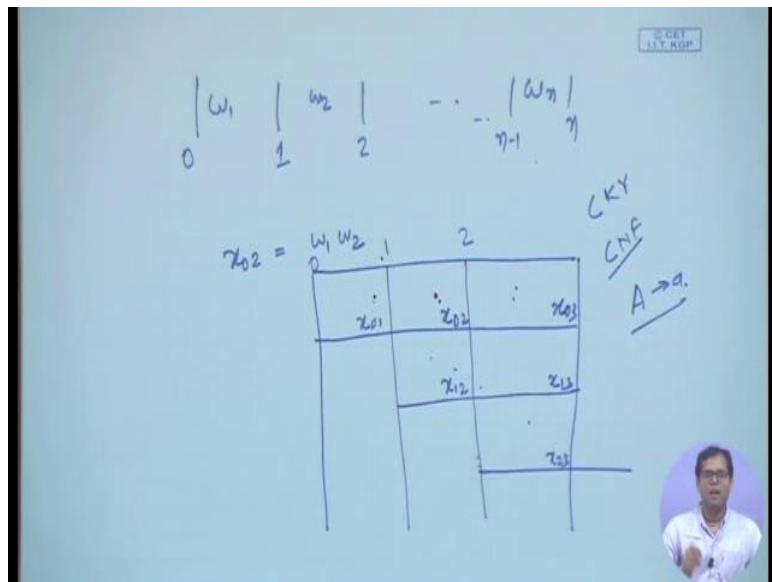
- Let  $n$  be the number of words in the input. Think about  $n + 1$  lines separating them, numbered 0 to  $n$ .
- $x_{ij}$  will denote the words between line  $i$  and  $j$ .
- We build a table so that  $x_{ij}$  contains all the possible non-terminal spanning for words between line  $i$  and  $j$ .
- We build the Table bottom-up.

A pink callout box labeled 'Home Exercise' contains the text: 'Use CKY algorithm to find the parse tree for "Book the flight through Houston" using the CNF form shown in the previous slide.'

At the bottom right is a circular profile picture of Prof. Pawan Goyal. The footer bar is dark blue with the text 'Pawan Goyal (IIT Kharagpur)', 'Syntax', and 'Week 3: Lecture 24'.

So, I will take a sentence; I have  $n$  words in my sentence. So, you will think about  $n$  plus one line that are separating them, is starting from 0 to 1. And so once you have done that any  $x_{ij}$  you will denote words between line  $i$  and  $j$ . So, you will build up a table such that any  $x_{ij}$  will contain all the possible non-terminal that can derive words between the lines  $i$  and  $j$ . And you do that bottom-up.

(Refer Slide Time: 01:23)



So, what I am saying, you have word 1, word 2 up to word n in your input stream. So, you assume some lines like 0, 1, 2, n minus 1, n. So, for example,  $x_{02}$  is words w1, w2 and so on. So, now what will you do? You build up a table and table will be some sort of triangular table. So, this can be 0, 1, 2; so this element will denote  $x_{01}$ , what are all the non-terminals that derive this word between 0 and 1. Similarly, here you will write  $x_{12}$  and so on. Suppose, they want a three words, this is  $x_{23}$ . So, we will write down all the non-terminals that derive this. You first fill this, then you will go next step,  $x_{02}$ ,  $x_{13}$ . And once you fill this, you will use  $x_{03}$  – the final one.

Now, what is the, where have we using the fact that this grammar is in CNF - Chomsky normal form, I will make sure the fact that at any point, this can come from only two non-terminals or a single terminal. So, at this point, when I am seeing as an individual word, so 0 1 will always be a individual word, similarly for 1 and 2. So, this will come from a rule capital A goes to small a, a rule of this kind. So, I will find out all the non-terminals that derives this terminal that is how I will fill the diagonal elements, the first diagonal elements.

Next time, this cannot come from a single terminal, because they are two words. So, this has to come from two non-terminals. So, I will find out if there is a rule that gives me these two non-terminals and that is why I am using Chomsky normal form. So, finally, when I am here, I will see if the sentences generate the whole sentence. So, I am catching all the possible intermediate steps here, so that is I have to do certain again and again.

So, this is the simple home exercise, so in the last lecture, we had given you a grammar in Chomsky normal form. Now, take the sentence, "Book the flight through Houston" and use the CKY algorithm to find the parse tree for that. So, but what I will do? I will do an example in today's class also so that it becomes more comfortable with using CKY algorithm.

(Refer Slide Time: 04:32)

CKY for CFG				
a 1	pilot 2	likes 3	flying 4	planes 5
DT	NP	-	-	S S
	NN	-	-	-
		VBZ	-	VP VP
			JJ VBG	NP VP
				NNS

$S \rightarrow NP VP$   
 $VP \rightarrow VBG NNS$   
 $VP \rightarrow VBZ VP$   
 $VP \rightarrow VBZ NP$   
 $NP \rightarrow DT NN$   
 $NP \rightarrow JJ NNS$   
 $DT \rightarrow a$   
 $NN \rightarrow pilot$   
 $VBZ \rightarrow likes$   
 $VBG \rightarrow flying$   
 $JJ \rightarrow flying$   
 $NNS \rightarrow planes$

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 3      4 / 14

So, let us take this example, a pilot likes flying planes and the grammar is given to you.

(Refer Slide Time: 04:46)

0 0 1 pilot 2 likes 3 flying 4 planes 5				
DT	NP:	-	-	$S \rightarrow NP VP$
NN:	-	-	-	$z_{23} z_{35}$
VBZ:	-	-	-	$z_{24} z_{45}$
VBG:	-	-	-	$z_{25} z_{5X}$
JJ:	-	-	-	$i_3 35X$
NP:	-	-	-	$i_4 45X$
VP:	-	-	-	
NNS:	-	-	-	
05 = 01 15   02 25   03 35   04 45				

So, let do that in the way I had explained, so 0, 1, 2, 3, 4, 5, there are five words. And this is the line. And now you should understand how do we denote these elements. So, this is  $x_0$  1, this is denote on we the first word, this is  $x_1$  2,  $x_2$  3,  $x_3$  4,  $x_4$  5. This will be  $x_0$  2 – words between 0 to 2.  $X_0$  3,  $x_0$  4,  $x_0$  5,  $x_1$  3,  $x_1$  4,  $x_1$  5,  $x_2$  4,  $x_2$  5 and  $x_3$  5. Now, I have to fill here, what are different non-terminals that filled each of the individual elements,  $x_0$  1.

What is  $x_0$  1? My word is a, so a pilot likes flying planes. So, is there a non-terminal that derives the word a. And if you see the grammar, DT derives a; so you can fill in DT here; DT derives a. Pilot, NN derives pilot. Likes, VBZ derives likes. Flying, so you there are two non-terminals VBG JJ they derive this; ‘and’ planes – NNS. So, filling the diagonal element is very easy, the first diagonal elements. Then you go the next step,  $x_0$  2. Now,  $x_0$  2 can come from  $x_0$  1 and  $x_1$  2, as a break up of these two points. So, is there an non-terminal where or is there a rule in my grammar when the right hand side, I have DT followed by NN. And if you see your grammar, yes, NP gives me DT followed by NN, so I can fill in NP here.  $X_1$  3 comes from  $x_1$  2 and  $x_2$  3, yes  $x_1$  3 pilot likes, so it comes from pilot and likes, so any non-terminal that gives me NN followed by VBZ. And if you see, there is no non-terminal. So, I will fill in empty here.

Similarly, and if VBZ followed by VBG, no; and if anything from VBZ followed by JJ,  $X_3$  5, VBG followed by NNS, yes, VP, and JJ followed by NNS – NP. So, there are two possibilities. So, fine this row is done, this diagonal is done. Now, we go to next step. Now, how do I derive  $x_0$  3 - a pilot likes. Now, that is why I am using the Chomsky normal form. I cannot derive it at a sequence of three non-terminals, because each individual non-terminal can give me at most, not at most exactly two non-terminals.

So, what are the two places, from which it can come? So, one possibilities I can break  $x_0$  3 as  $x_0$  1 and  $x_1$  3; one word into two words or  $x_0$  2,  $x_2$  3, there are two possibilities. So, I have to check individually each of these possibilities,  $x_0$  1  $x_1$  3;  $x_0$  1 is DT followed by null. So, this is already gone; this is no non-terminal.  $X_0$  2 is NP followed by, 2 3 is VBZ, so NP followed by VBZ. Is there a non-terminal, when the right hand side I have NP followed by VBZ. And if you see your grammar, there is nothing. So, this is also null. So, here it is null, there is no non-terminal let me derive, a pilot likes.

Now, pilot likes flying, 1 to 4. So, again 1 to 4 will be 1 to 2, 2 4, or 1 3, 3 4. So, 1 2, 2 4, 1 2 is NN, 2 4 is null – so this part is null. 1 3, 3 4, 1 3 is null, so this is becomes 1 also. 2 5 will

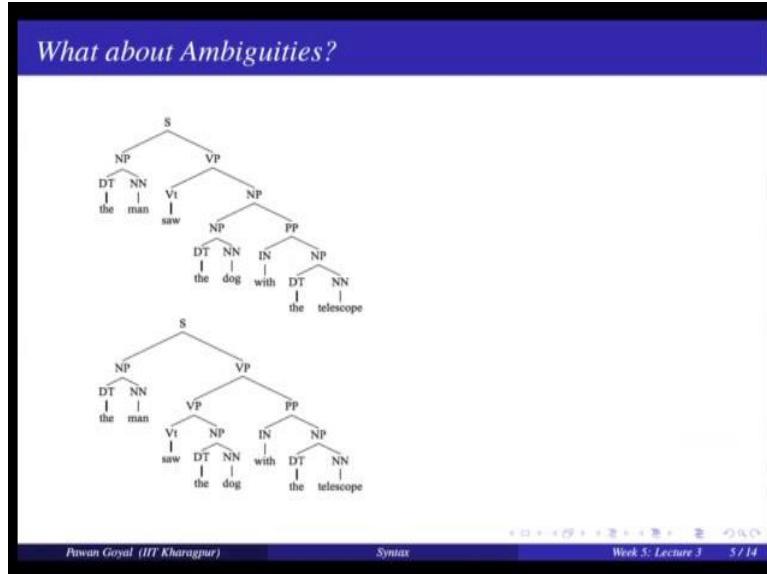
be 2 3, 3 5, and 2 4, 4 5. So, 2 3 here is VBZ, and 3 5 is VP. Is there something that VBZ and VP, yes, VP this means VBZ and VP. 2 4, 4 5; 2 4 is null; so this is my VP. Similarly, now you will go to x 0 4. Now, what are difference which you can break x 0 4. Now, 0 4, again you to break in a sequence of two non-terminals, so it can be 0 1 followed by 1 4 or 0 2 followed by 2 4 or 0 3 followed by 3 4, there are three ways.

So, let us see one-by-one. 0 1 is DT, 1 4 is null, so this part is gone. 0 2 is empty and 2 4 is null, this part is also gone, 0 3 is null already, so this is null. 1 5, 1 5 can be 12 25, 13 35, 14 45. 12 and 25, yes, NN followed by a VP, so is this something in my grammar, NN followed by VP, no, so this is gone. Similarly, 13 followed by 35; 13 is null, and 13 is null, so this is gone. 14 followed by 45; 14 is null, so this is also gone, so this is also null. Now, the only thing remain is x 0 5.

Now, how can I fill x 0 5, what are different ways. So, that let me write down here. 05 can be 01 followed by 15; 02 followed by 25; 03 followed by 35; 04 followed by 45. 01 followed by 15; 01 is there, 15 is null, this is gone. 02 followed by 25; 02 is there and 25 is also there; this is NP followed by VP. And this is a sentence in my grammar. So, S gives me NP VP, so this is one possibility already, so; that means, this sentence is grammatical at least. There is one S that derives this, but are there any further S.

So, for that we have to look at other possibilities 03 35, this is null; and 04 45 is null, so fine. So, if you see there, I think we made one mistake here; there should have been another VP in this case. So, there is VP 1, VP 2. And then there will be S NP VP 1 and N P VP 2. So, I will search that look back into this calculation we did and see where we made a mistake. But everything else is the same that we did here. So, there are two different S in which this sentence can be parse tree, two different ways. So, now once you know this, I will say that use the previous example so that is Book the flight through Houston from the other grammar and try to get its parse tree.

(Refer Slide Time: 14:09)



So, now in this example, what we saw, there are two possibilities. So, can you think of the possibilities, why there are two possibilities in this case, a pilot likes flying planes. So, whether he likes to fly the plane or whether he likes to see flying planes something like that, so there are interpretation, that is why there are two different parse edge of this sentence. So, each individual parse will denote one particular interpretation. Now, by using this context by this CKY algorithm, we denote all the possible parse trees using my grammar. But I have no way of saying which parse is more probable than the other parse. I cannot assign some probabilities to them. So, something if I have so this is the sentence, the man saw the dog with the telescope, it has two different interpretation in terms of two different parse edges. Whatever I have covered till now, it cannot tell me which parse is more probable than other.

(Refer Slide Time: 15:15)

*Probabilistic Context-free grammars (PCFGs)*

**PCFG:**  $G = (T, N, S, R, P)$

- $T$ : set of terminals
- $N$ : set of non-terminals
  - For NLP, we distinguish out a set  $P \subset N$  of pre-terminals, which always rewrite as terminals
- $S$  : start symbol
- $R$ : Rules/productions of the form  $X \rightarrow \gamma$ ,  $X \in N$  and  $\gamma \in (T \cup N)^*$
- $P(R)$  gives the probability of each rule.

$$\forall X \in N \sum_{X \rightarrow \gamma \in R} P(X \rightarrow \gamma) = 1$$

Ptwan Goyal (IIT Kharagpur)      Syntax      Week 5: Lec 1

So, now we would like to have a way in which we can assign them the probabilities that this parse edge is more probable than the other. And for that what we use is called probabilistic context free grammars. So, this is simple extension of context free grammar, where in addition to whatever we have seen in context free grammar, each rule is also signed some probability. So, as you see in the formulation, it is  $T$ ,  $N$ ,  $S$  and  $R$ . They are exactly same as what we had seen in the context free grammar plus there is something called  $P$ .

$$\forall X \in N \sum_{X \rightarrow \gamma \in R} P(X \rightarrow \gamma) = 1$$

So, I am assigning a probability distribution over the rules, and only the constraint here is that from a given non-terminal the lateral side the probability generating anything should add up to 1. So, if there are 5 possibilities, if I add all the 5 possibilities, the rule for all the 5 possibilities, they should add up to 1. So, this is the constraint. So, probability in the rule gives the probability of each rule  $P(R)$ , so the constraint is for all  $X$  in non-terminals, probability of  $X$  to  $\gamma$  for all the possible  $\gamma$ s should add up to 1.

(Refer Slide Time: 16:31)

A Simple PCFG (in CNF)			
S	→	NP VP	1.0
VP	→	V NP	0.7
VP	→	VP PP	0.3
PP	→	P NP	1.0
P	→	with	1.0
V	→	saw	1.0
NP	→	NP PP	0.4
NP	→	astronomers	0.1
NP	→	ears	0.18
NP	→	saw	0.04
NP	→	stars	0.18
NP	→	telescope	0.1

Pawan Goyal (IIT Kharagpur)

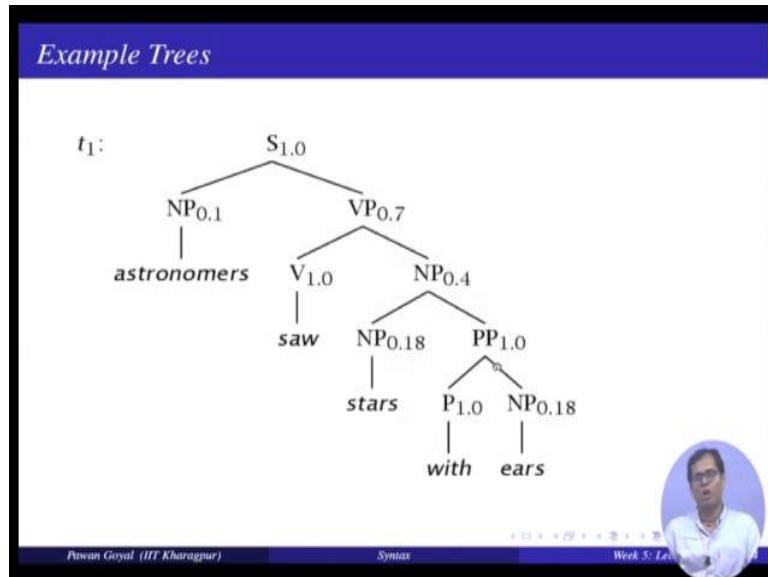
Syntax

Week 5: Lecture 3 7/14

So, let me give you an example. So, this is one simple CNF, sorry simple PCFG in Chomsky normal form. So, what do you see here? From S, there is only one rule, S goes to NP VP. So, because there is only one rule, it has a possibility of 1. From VP, there are two rules; VP can give, V followed by NP or VP followed by PP. So, the constraint is that for the possibilities of these two rules should add up to 1, so that is what is happening here. The first rule has a possibility of 0.7; second rule has a possibility of 0.3, these two add up to 1. Now, PP gives me P NP, only one rule with PP on left hand side, so this is the possibility of 1. P gives me only with again possibility 1; V gives me saw, this is the possibility 1.

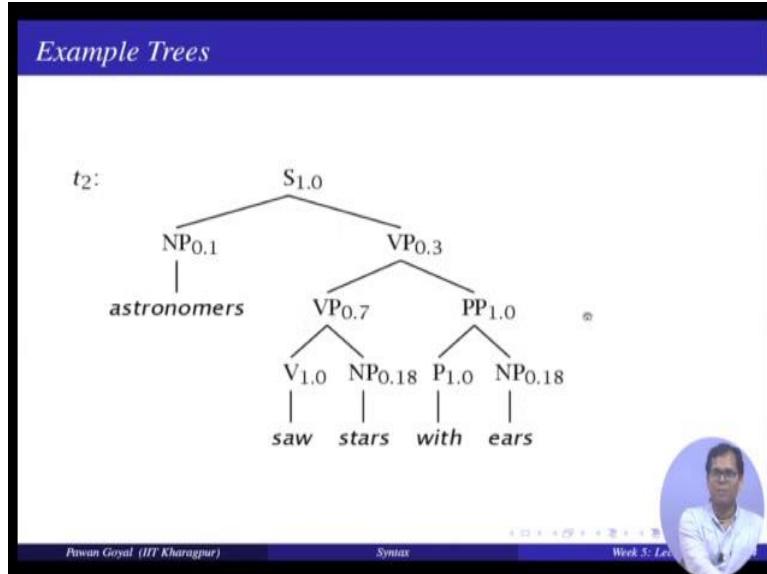
Now, all these rules in the right hand sides are starting from the NP. NP giving to me NP PP, all the words let us call as like astronomers, ears and so on. So, all these should add up to 1. And you can see that this actually happen 0.4 plus 0.1 – 0.5, 0.68, 0.72, 0.9 and 1. So, all these possibilities are adding up to 1. So, this is the constraint that is being followed. The rules have the same format as in context free grammar, but you shall have possibilities. Now, how does it help? It helps in that I can now assign possibilities to each individual parse tree.

(Refer Slide Time: 18:03)



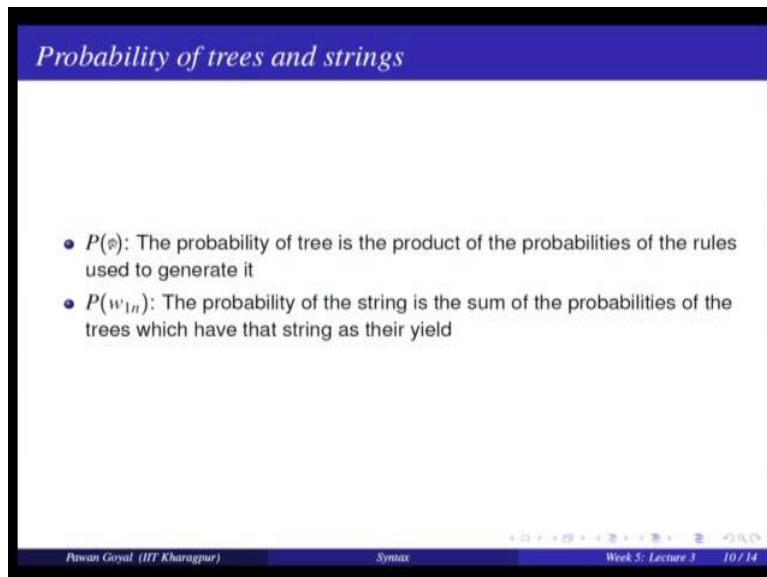
So, suppose this is one parse tree. Astronomers for the sentence astronomers saw starts with ears. How do I find the probability of this parse tree? This is the parse. So, I know S gives me NP and VP, yes this is the first rule that I am applying. Now, as my PCFG, the probability of this rule is one. S giving NP, VP is 1, so I have this one here. NP giving astronomers, deriving astronomers is probability 0.1. VP deriving V and P is probability 0.7; V giving saw is 1. NP giving NP, V P is 0.4; NP giving stars is 0.18 and so on. These are the rule probability as per my grammar as my PCFG. So, what I will do? I will just multiply all these probabilities 1 times 0.1 times 0.7 times 0.1 times 0.4 times 0.18 times 0.18, this is my probabilities of this parse trees.

(Refer Slide Time: 19:03)



And if I get a second parse tree, where instead of VP giving me V and NP, VP giving VP followed by PP, I can again compute its probability by multiplying its corresponding rule probabilities. And I can find the probabilities of both the parse trees individually.

(Refer Slide Time: 19:21)



So, now how do I use that to compute the probability of the tree that is simply the product of the probabilities of all the rules that I used to generate this? And probability of assignments and the probability of sentences is nothing, but find out all the parse trees and the probabilities of the individual parse trees and just sum them up. So, probability of this

sentence is the sum of the probabilities of the trees that have this as their yield that is another way of saying that those parse trees are used to generate this particular strings.

(Refer Slide Time: 20:05)

*Tree and String probabilities*

$$w_{15} = \text{astronomers saw stars with ears}$$

$$\begin{aligned} P(t_1) &= 1.0 * 0.1 * 0.7 * 1.0 * 0.4 * 0.18 \\ &\quad * 1.0 * 1.0 * 0.18 \\ &= 0.0009072 \end{aligned}$$

$$\begin{aligned} P(t_2) &= 1.0 * 0.1 * 0.3 * 0.7 * 1.0 * 0.18 \\ &\quad * 1.0 * 1.0 * 0.18 \\ &= 0.0006804 \end{aligned}$$

$$\begin{aligned} P(w_{15}) &= P(t_1) + P(t_2) \\ &= 0.0009072 + 0.0006804 \\ &= 0.0015876 \end{aligned}$$


Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 3

So, in the first case, I had this sentence, and they were to parse trees, I can compute the probabilities of individual one. So,  $P(t_1)$  and  $P(t_2)$ , I can find out and to find the probability of the whole sentence, I just add up these two probabilities  $P(t_1)$  plus  $P(t_2)$  and that gives me the probability of this whole sequence.

(Refer Slide Time: 20:27)

*"Book the dinner flight"*

*Probabilities*

- Parse tree 1:  $.05 \times .20 \times .30 \times .20 \times .60 \times .20 \times .75 \times .10 \times .30 = 1.62 \times 10^{-6}$
- Parse tree 2:  $.05 \times .05 \times .30 \times .20 \times .60 \times .75 \times .10 \times .15 \times .75 \times .30 = 2.28 \times 10^{-7}$

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 3      12 / 14

Similarly, if I have another sentence like book the dinner flight and as per the different grammar, I can compute the parse trees - the two parse trees. Compute the probabilities for the individual one. So, parse tree 1 here, book the dinner flight is  $1.62 \times 10^{-6}$ . And book the dinner flight will have a probability of  $2.28 \times 10^{-7}$ . So, one thing I can immediately see is that the first one parse is more likely interpretation than the second one.

(Refer Slide Time: 20:59)

*Features of PCFGs*

- As the number of possible trees for a given input grows, a PCFG gives some idea of the plausibility of a particular parse
- *But* the probability estimates are based purely on structural factors, and do not factor in lexical co-occurrence. Thus, PCFG does not give a very good idea of the plausibility of the sentence.
- Real text tends to have grammatical mistakes. PCFG avoids this problem by ruling out nothing, but by giving implausible sentences a low probability
- In practice, a PCFG is a worse language model for English than an n-gram model
- All else being equal, the probability of a smaller tree is greater than a larger tree

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 3      13 / 14

Let us look at some of the features of probabilistic context free grammars. So, why we started with this formulation? So we said that using the context free grammar, given a sentence we can find out all the possible parse trees, but you are not able to assign any probabilities to that. So, by using PCFG, if for a given string, the number of parse trees are increasing, I can also assign the probabilities for each individual parse trees, so that gives me some plausibility, which parse tree is more probable than the other one for the given string. So, this is important, but at the same time we should understand that although by using PCFG, I can compute what is the probability of the sentence by taking all the parse trees, taking the individual probabilities and adding them up, this is just not giving a very good plausibility of the sentence. This is only looking at the structural factors not some lexical co-occurrence.

So, in general, to find the probability of the sentence, you would prefer to use language model than a PCFG. PCFG is good only for finding the probability of a parse tree, which parse is

more probable than another one. Yes, if it helps in some cases like in real text you might find some grammatical mistakes, so PCFG will allow that, but will give you very, very low probability. So, in one case, you can also probably find out which sentence has some grammatical mistake. If the PCFG is giving it low probability. And yeah, this is something that I said earlier; so in practice, this is not good for modeling the probability of the sentence, language model is much better than PCFG.

So, why is that the case, so a simple example is if we have the same sentence, I have two different trees, one is smaller than the other, the smaller tree will always have a higher probability than the larger one, because all the probabilities are less than one. And for a larger tree, you are multiplying the probabilities. So, to take away here is that, you would use PCFG to find out what are all the probabilities for different parse trees for a sentence and try to choose the best one.

(Refer Slide Time: 23:24)

*Important Questions?*

Let  $W_{1m}$  be a sentence,  $G$  a grammar,  $t$  a parse tree

- What is the most likely parse of sentence?

$$\text{argmax}_t P(t|w_{1m}, G)$$

- What is the probability of a sentence?

$$P(w_{1m}|G)$$

- How to learn the rule probabilities in the grammar  $G$ ?



Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 1

$\text{argmax}_t P(t|w_{1m}, G)$

So, now once we have this formulation of PCFG, there are some interesting questions that we would like to explore using that. So, suppose I am given a sentence  $W_{1m}$ , I am given a grammar  $G$ , and there are various parse trees  $t$  is one of those. So, what is the most likely parse for this sentence given the grammar? So, which parse is tree  $t$  gives me maximum probabilities argmax of probability given the sentence and the grammar. Then what is the probability of the sentence? Probability of the sentence given by grammar, and then finally,

how do we learn the rule probabilities of my grammar  $G$ , which these are the three interesting questions that we would like to answer in the next lectures.

So, for example, how do I find the most likely parse of a sentence? So, one simple solution is find out all the possible trees and take the one with the highest probability, but is there any efficient method for doing that. What is the probability of a sentence? Again I can find out probabilities of the individual parse trees add them up, this gives me the probability of the sentence, but can I do a t extortion numerating all that and adding it or is there any some other methods. And finally, there is some interesting question that how do I learn the rule probabilities in the grammar  $G$ . And the answer to this will be similar to what we saw in the case of Head and Markov models.

So, there are they will be again two ways of running the parameter; one will be when the I am given the corpus and some labeled parse trees, I can use them to find the probabilities. Under the scenario, where I am given the corpus, but not the parse trees, I am only given the grammar, grammar in the sense CFG, not the rule probabilities, then how do I learn the parameter. So, this will be again very, very interesting topic and there you can use some ideas where that I had shown earlier for the (Refer Time: 25:33) algorithm. So, in the next lecture, we will start trying to answer some of these questions.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 25**  
**PCFGs – Inside-Outside Probabilities**

Hello everyone, welcome back to the 4th lecture of this week. In last lecture, we had discussed what is PCFGs and we finally, came up with what are the interesting problems that we would like to answer. So, for example, what is the most likely parse for a given sentence as per my PCFG grammar and what is the probability of the sentence as per the grammar? Finally, we also wanted to see how do I learn my PCFG rule probabilities. If you remember, this was the only difference between context free grammar and the probabilities version is that all the production rules here have a probability value.

So, how do I learn these values? So, this is what we will be covering in this lecture and in the next lecture and we will be using a specific concept called inside outside probabilities and this will be very analogous to what you study in forward backward algorithm in terms of learning parameters for HMM. So, we will go to that.

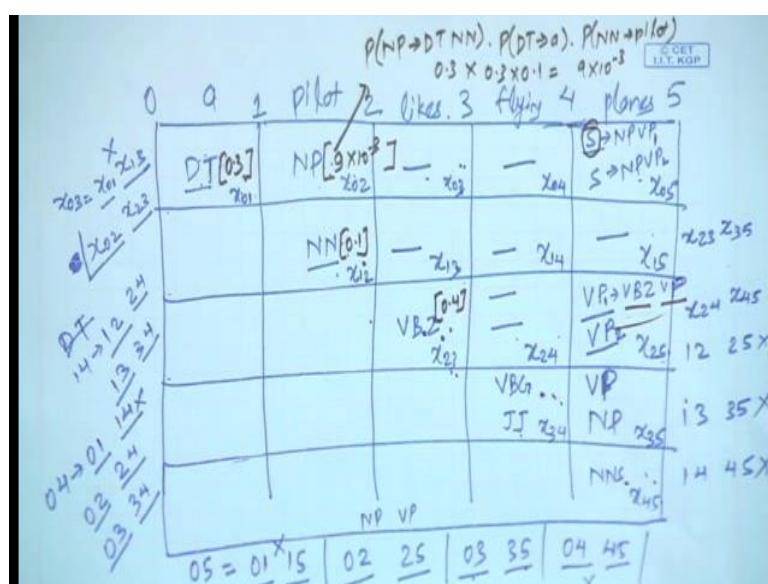
Starting with the first problem, how do I find out the most likely parse for a sentence? And this can be solved simply if we use the CKY algorithm for PCFG. So, from the example that we did in the last class, I hope you understand now how to use CKY algorithm for the context free grammar. So, today I will show, how do you extend that for PCFG?

(Refer Slide Time: 01:43)

a 1	pilot 2	likes 3	flying 4	planes 5	
DT [0.3]	NP [.009]	-	-	S [1.4688x10 <sup>-5</sup> ] S [6.12x10 <sup>-6</sup> ]	$S \rightarrow NP VP$ $VP \rightarrow VBG NNS$ [0.1] $VP \rightarrow VBZ VP$ [0.1] $VP \rightarrow VBZ NP$ [0.3] $NP \rightarrow DT NN$ [0.3] $NP \rightarrow JJ NNS$ [0.4] $DT \rightarrow a$ [0.3] $NN \rightarrow pilot$ [0.1] $VBZ \rightarrow likes$ [0.4] $VBG \rightarrow flying$ [0.5] $JJ \rightarrow flying$ [0.1] $NNS \rightarrow planes$ [.34]
	NN [0.1]	-	-	-	
		Vbz [0.4]	-	VP [.001632] VP [.00068]	
			JJ [0.1] VBG [0.5]	NP [.0136] VP [.017]	
				NNS [.34]	$0.009 \times 0.00068 \times 1.0 = 6.12 \times 10^{-6}$

Let us take an example. So, we have this PCFG. So, what you are seeing here with each rule, you have a rule probability as well and you can verify that all the probabilities starting from a given right hand side will add up to 1. So, now, I want to find out what is the most likely parse for the sentence, the same sentence the pilot likes flying planes. So, for that I need to find out what is the probability with which I can find a non terminal as in the final position for deriving this whole sentence 0 to 5. So, we proceed nearly in the same manner as we did in the case of CKY. So, what I will do? I just take the same example, how we solve CKY in addition and what are the things that you need to do different here.

(Refer Slide Time: 02:50)



This is what we had seen last time. So, if we have to build, if we have to find out, what is the; whether the sentence can derive this whole thing, I will fill it starting from 0 1, 1 2, 2 3, 3 4, 4 5, then I will go further. So, now, I will just tell, what will differ in the case of PCFG. So, now, with each non terminal, there will be probability with which it can derive that particular terminal or a set of terminals. So, it is starting from the first element. So, this says that the non terminal DA derives a. Now I will write down here, what is the probability? So, if you go to your grammar, the probability DT gives me is 0.3. So, I will put 0.3 here.

Similarly, for this element, what is the probability that NN gives me pilot and if I look at my grammar, this is 0.1, similarly here what is the probability that VBZ derives likes this gives me 0.4 and like that I will fill all these entries now this is ok.

Now, what I want to show you is that how do I fill the next entry that is 0 2, what is the probability with which the non terminal NP derives the sequence a pilot? Now if you remember, a single non terminal can derive 2 non terminals; 2 terminals only via 2 non terminals. So, it has to first derive x 0 1, x 1 2 then each of these will individually derive a N pilot. So, how do I fill in this probability value? So, this would be probability that NP derives DT NN times probability DT derives a times probability, NN derives pilot. This probability I can get from my PCFG. So, this probability if I see in my grammar, this is 0.3, this probability I have already filled in is 0.3 and this probability also I have filled in this is 0.1.

The probability that NP derives a pilot here comes out to be 9 times 10 to the power minus 3 and that is what I fill in here and in the same manner you will fill all the entries if it is 5, it means the probability will be 0 and if there is a non terminal here; that means, there will be certain probabilities and this you can get by finding out probability individually for each of this and multiplying the probability of this rule. So, that way you can incrementally build all the stable from in a bottom up fashion we are starting from the first element then in the next one and so on. So, remember nearly same as we did in the last class only the probabilities and the multifunctional probabilities will change, suppose you do that for this example, this is what you will find out. So, these are the rule probabilities that you will come up with. So, I will encourage all of you that you should try this on your own and see that you can get the same set of values that have been shown in the slide.

(Refer Slide Time: 06:35)

*Probability of a String*

$P(w_{1:m}|G)$

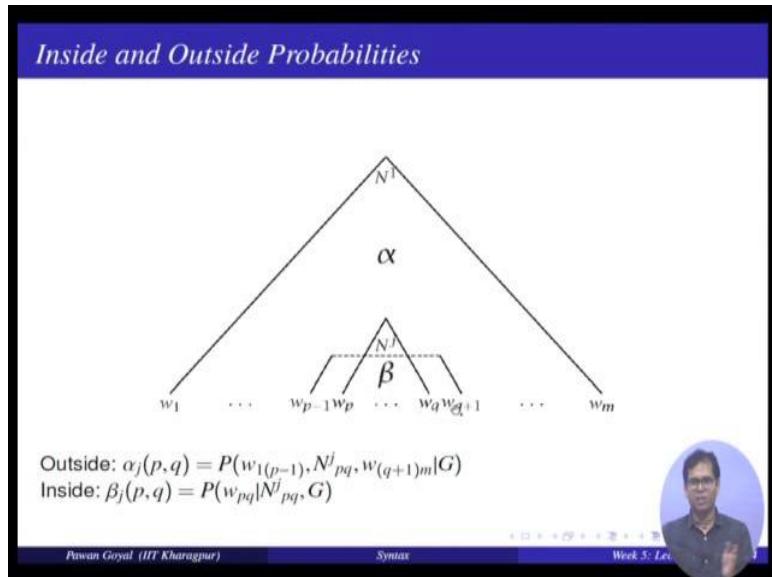
- In general, simply summing the probabilities of all possible parse trees is not an efficient way to calculate the string probability
- We use *inside algorithm*, a dynamic programming algorithm based on inside probabilities.

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 4      4 / 14

Now coming to the next question, I am given the PCFG as my grammar, how do I find out, what is the probability of the sentence. So, one simple solution you can always find is that I find out all the possible parse trees, find the probabilities for each of the parse trees and add those up that is one way, but this you might be inefficient if there are very huge number of parses trees, if you remember one of the introductory slides, we said that as you increase the number of words in the sentence or the number of phrases, the number of possible parses increase roughly exponentially, there was something like Peclet number with this their group and it can go to 100 plus parses.

So, you do not want to sum this get the probability individually and then do addition you want to do something more efficient than that and how do we do this and that is why we use this inside algorithm that is one of the main topics of this lecture, so what is the inside algorithm. And this is some dynamic programming algorithm based on the concept of inside outside probabilities.

(Refer Slide Time: 07:41)



Now I will try to introduce, what is that inside outside probability. So, this is very central to understanding this lecture as well as the next lecture. So, let us try to understand; what is the concept and just so that it helps you, this is very very similar to the concept of forward backward algorithm that we did in the case of HMFs. So, what is inside outside probabilities? So, I can this is parameterized for certain node of my tree. So, what you see in this figure, I have a sentence with words W 1 to W m and N 1 is the non terminal you can think of it as the sentence non terminal that derives the whole sequence of words using my grammar.

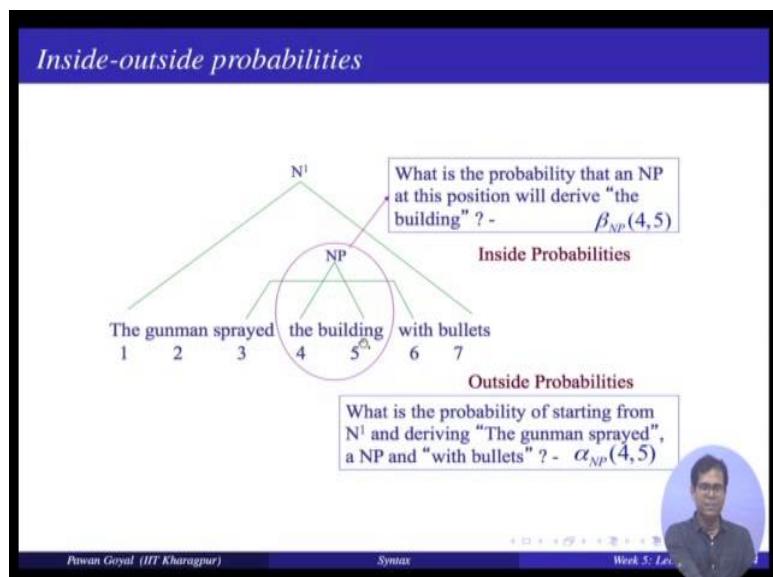
Now, I am putting a parameter here that is N j; N j is one particular non terminal. It can be N p v p or so on. So, what I am saying here assume that there is a non terminal N j that derives the sequence W p to W q, again p and q are arbitrary, you can take any p starting from 1 to m minus 1. So, what I am parameterized here is that this N j derives the sequence W p to W q. So, now, with respect to that I am defining, what is my inside probabilities and what is my outside probabilities. So, you can as such you can guess this is something outside of that and this is something inside of that.

How do I define my inside and outside probabilities? So, outside probability is that it is starting from N 1, I can derive the sequence W 1 to W p minus 1, I can derive this non terminal W sorry, N j and the sequence W q plus 1 to W m, this is my outside probability and inside probability is given this N j the probability that I can derive W p to W q as per my grammar and it will then follow the same sort of ideas that if I can multiply inside and outside

probability that will give me something like the probability of the sentence parameterized by this particular  $j$  th non terminal. So, formally that is how I can write the outside and inside probabilities.

Outside probability -  $\alpha_{N^j} p q$  that is standard format which we can write  $p q$  denotes the sequence  $W^p W^q$  on which this parameterized and  $j$  denotes the particular non terminal  $N^j$  here. So, the outside probabilities, probability of generating  $W^1$  to  $W^p$  minus 1 s and  $j p q$  and  $W^q$  plus 1 to  $m$  given by grammar and inside probabilities probability of generating  $W^p$  to  $W^q$  given  $N^j$ , so this is also parameterized by  $N^j p q$  because it is arriving  $W^p$  to  $W^q$  and in my grammar. So, that is why I define my inside and outside probabilities.

(Refer Slide Time: 10:42)



Now, let us take a simple example, I have the sentence, the gunman sprayed the building with bullets and here your  $p q$  is 4 and 5. So, you are parameterization with respect to this sequence the building. So, accordingly this will be particular non terminal that is NP that is deriving the building. So, now, what do we mean by inside probability? So, inside probabilities; what is the probability that this non terminal NP at this location derives the sequence the building, this is my inside probability.

(Refer Slide Time: 11:38)

### Inside-outside probabilities

$\alpha_{NP}(4,5)$  for "the building"  
=  $P(\text{The gunman sprayed}, NP_{4,5}, \text{with bullets} | G)$   
 $\beta_{NP}(4,5)$  for "the building" =  $P(\text{the building} | NP_{4,5}, G)$

```
graph TD; N1[N^1] --> [The gunman sprayed]; N1 --> NP[NP]; [The gunman sprayed] --> 1[1]; [The gunman sprayed] --> 2[2]; [The gunman sprayed] --> 3[3]; NP --> theBuilding["the building"]; NP --> withBullets["with bullets"]; theBuilding --> 4[4]; theBuilding --> 5[5]; withBullets --> 6[6]; withBullets --> 7[7]
```

Ptwan Goyal (IIT Kharagpur) Syntax Week 5: Lecture 4 7 / 14

And what will be the outside probability here that is starting from N 1, I can derive the sequence, the gunman is sprayed this NP and with bullets this is my outside probability. So, this is what we have defined the outside probability alpha NP, 4 5 would be the probability of this sequence the gunman is sprayed NP 4 5 and with bullets given my grammar and the inside probability beta NP 4 5 would be the probability for deriving the building given NP 4 5 and grammar.

Now, this is the definition of inside and outside probabilities. Now how do I actually compute this inside and outside probability given this sentence and suppose our grammar is also given to me how do I compute that?

(Refer Slide Time: 12:20)

*Inside Probabilities: Base Step*

$$\beta_j(p, q) = P(w_{pq} | N^j_{pq}, G)$$

*Base case*

$$\begin{aligned}\beta_j(k, k) &= P(w_{kk} | N^j_{kk}, G) \\ &= P(N^j \rightarrow w_k | G)\end{aligned}$$

*Base case for pre-terminals only*

E.g., suppose  $N^j = NN$  is being considered and  $NN \rightarrow building$  is one of the rules with probability 0.5

$$\beta_{NN}(5, 5) = P(building | NN_{5,5}, G) = P(NN_{5,5} \rightarrow building | G)$$

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 4      8 / 14

$$\beta_j(k, k) = P(w_{kk} | N_{kk}, G)$$

$$= P(N^j \rightarrow w_k | G)$$

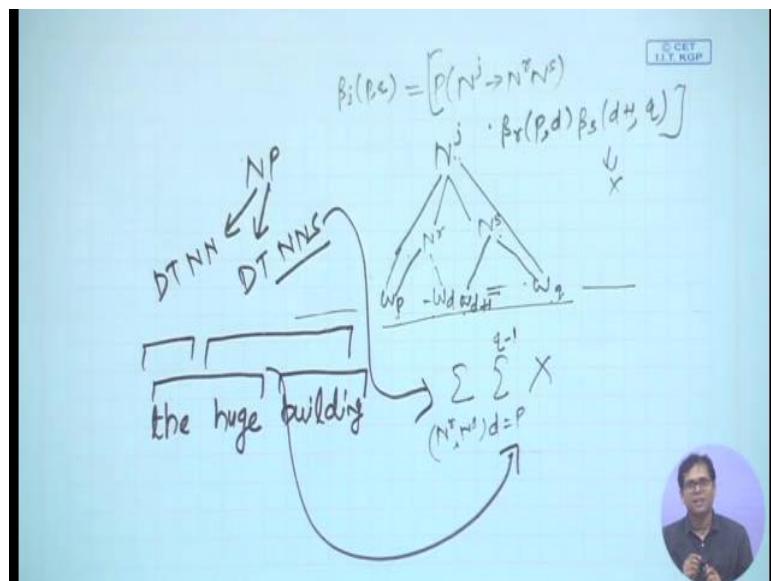
For inside probability, so let us see this is the definition of inside probability  $\beta_j(p, q)$  is probability of deriving the sequence  $W_p$  to  $W_q$  given non terminal  $N_j$   $p$   $q$  in my grammar now. So, I will be deriving both these probabilities in iterative manner. So, what will be the base case here? So, the base case here if you see would be when  $p$  and  $q$  are same that is I am deriving a single terminal from a non terminal and that you can find out very easily if your grammar is in the Chomsky normal form. So, what is the particular non terminal that derives this term? So, this would be the base case for deriving a single term base and then I will go in a bottom up fashion derive the inside probabilities for higher and higher sequences. So, the base case here is when  $P$  is equal to  $q$ . So,  $\beta_j(k, k)$  would be probability of  $W_k$  given  $N_j$   $k$   $k$  and  $G$ . Now  $W_k$  is a single word sequence from  $k$  to  $k$  and I can immediately find out what is the role in my grammar that derives this word  $W_k$  and I put this rule probability here and this will give me the this probability  $\beta_j(k, k)$ .

Now, as in example, suppose here my word is *building*. So, this is the 5th word. So, I am computing beta for 5 5 and suppose I take  $N_j$  is equal to *NN* that works in non terminal derives this word. So, you can compute beta *NN* 5 5 as the probability with which the non

terminal NN can derive this word building and this is easily given by my grammar. So, this is my base case.

Now, what will be the inductive case? So, I have to go bottom up. So, let us take a generic beta j p q how do I derive it in terms of the smaller values. So, let us try to do that.

(Refer Slide Time: 14:32)



If I am given I have to compute beta j p q, what is that mean? It means that I have a non terminal N j and that is deriving a sequence W p up to W q and I have to compute this probability given N j the probability of the sequence W p to W q because I am conducting in a I am doing it in a bottom fashion and I have to use the proper lower values.

I will first say N j can give me some N r and N s, now N r will derive from W p to some W d. So, very generic case, so this can be any possible N r and N s that my N j can derive and again I can go from W p to any W d, so you can see what can d vary from to so, d can vary from P up to q minus 1 and this N s gives me W d plus 1 up to W q this is my recursive case. Now how do I write down this probability? So, I will say this is first the probability that N j derives the rule the non terminal the N r and N s.

Probability N j derives N r N s times this probability now here N r is given. So, what is the probability is that N r given N r; it derives from W p to W d again, you can compute this using. So, you can express this using inside probability that is beta r p d, similarly this 1 beta s d plus 1 to q, now this is for the particular variation that I have shown, but in general, d can

vary from p to q minus 1 and they can be any possible N r and N s. So, I will have to write down this whole thing, let me call it X. So, I have to say this can vary from p to q minus 1 and all possible N r N s and I will put x here. So, this is my inductive step.

(Refer Slide Time: 17:13)

### Inside Probabilities: Induction Step

Assuming Chomsky Normal Form, the first rule must be of the form  $N^j \rightarrow N^r N^s$

$$\beta_j(p, q) = \sum_{r,s} \sum_{d=p}^{q-1} P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)$$

- Consider different splits of the words - indicated by d  
E.g., *the huge building*
- Consider different non-terminals to be used in the rule:  
E.g.,  $NP \rightarrow DT\ NN$ ,  $NP \rightarrow DT\ NNS$

Navigation icons: back, forward, search, etc.

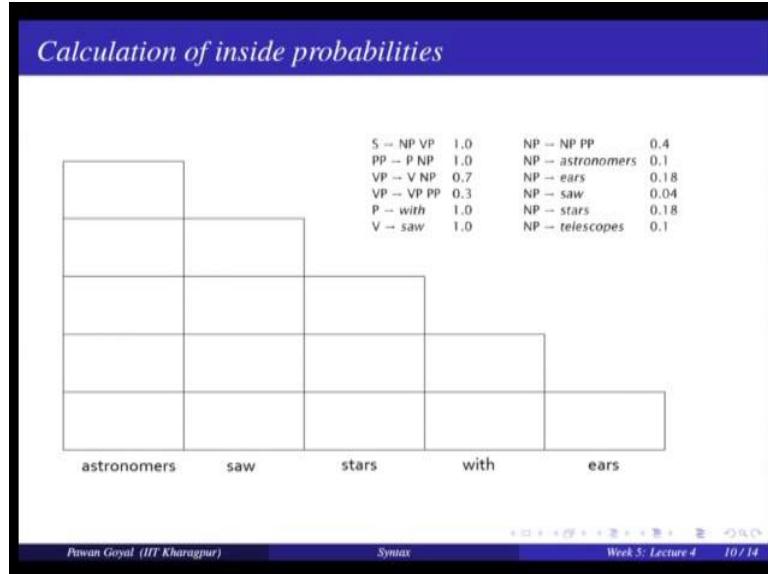
Piwan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 4      9 / 14

$$\beta_j(p, q) = \sum_{r,s} \sum_{d=p}^{q-1} P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)$$

Now, what do I mean by parameterization over by all possible r s and all possible d. So, if you take the case of this simple phrase, the huge building. So, what do I mean by various d? So, if I have this is the huge building by variation of d, I mean whether I am composing it by the huge and building or the and huge building 2 possible variables of d and what do I mean by parameterization over all possible N r and N s. So, the particular non terminal that derives this whole thing can be support this NP, it can come via say NP gives me DT NN, this is 1 possibility, this is DT NN or there can make another rule, NP gives me DT NNS. This is another possibility. So, this is the parameterization over all possible N r N s and this is over out possible d's.

Now we have seen that you can compute it for the only the terminals first and then the bottom up to compute it for other values.

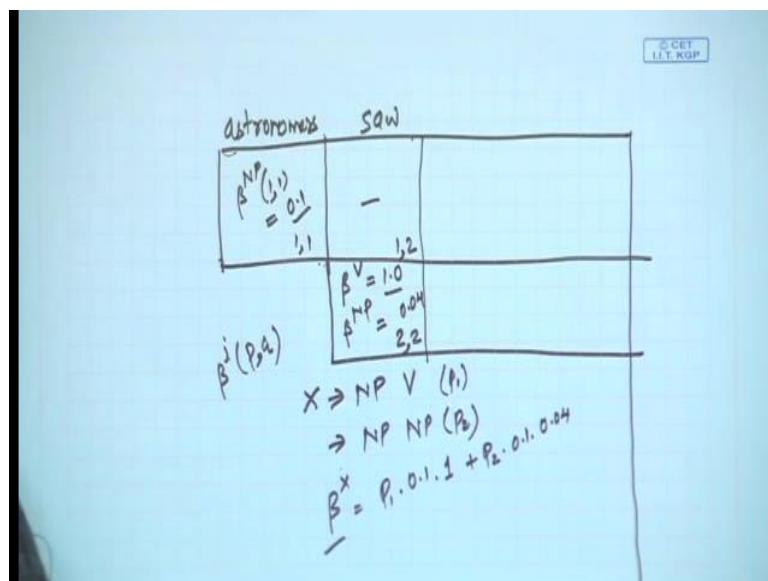
(Refer Slide Time: 18:40)



Now if you do a simple example, so this is again similar to what we have seen earlier. So, I am given this PCFG and this is my sentence and I want to compute the inside probabilities here. So, how do I go about out it and you looking at the table you can see that we will doing something very similar to what we did in the case of CKY algorithm.

But now we will focus on filling up what is the inside probability of deriving this from the particular non terminal. So, let us just try to filling some entries here.

(Refer Slide Time: 19:36)



Let us try only the first 2, this is astronomers and this is saw and I have to fill in the inside probabilities. So, this would be, so now, in the way we defined inside probability this would be 1 1, this would be 2 2 sequence from word 2 to word 2, this would be sequence from word 1 to word 2.

Now when I define beta j p q, so, here p and q are both 1. So, I have to find out what is the non terminal that derives these word astronomers, now if I see a grammar and these derive astronomers with the probability of 0.1. So, I can fill the inside probability here beta NP 1, 1; 1, 1, you need not to write and I am just writing here is equal to 0.1 because 1 1 is already defined by the particular cell where we are, similarly let us go to 2, 2; 2, 2, is saw if I see the grammar NP derives saw NP derives saw. So, there are 2 different. So, that is where you can see there are 2 different j that are possible here. So, one is beta V, another is beta NP and this has the probability of one and this has the probability of 0.04 using my grammar.

Now, suppose I have to fill this one 1 2, what is the particular non terminal that will derive this whole thing again similar to what we were doing in CKY is the something that derives NP and NP if I see my grammar nothing derives NP and NP in the right hand side. Now is there something that derives NP followed by V, again there is nothing here NP followed by V. So, this will come out to be 0. Suppose there was something that derives NP followed by V. So, I would simplify find out what is the non terminal that is deriving this and put the probability as this times this times the rule probability.

Similar to what we were doing in the case of CKY for probability context free grammars the only difference here is that if there are multiple splits for same non terminal, I will add all these. So, for example, the same non terminal so I am taking a hypothetical case, suppose the same non terminal X derives NP followed by V and also NP followed by NP if the probability of P 1 is the rule probability, this is a probability of P 2. So, I will put here beta x and this will be P 1 times 0.1 times 1 plus P 2 times 0.1 times 0.04 and that is why I fill all the entries.

I will take care of all the possible ways in which this particular non terminal, we can derive this.

(Refer Slide Time: 23:41)

Calculation of inside probabilities				
1	2	3	4	5
$\beta_{NP} = 0.1$		$\beta_S = 0.0126$		$\beta_S = 0.0015876$
		$\beta_{VP} = 0.126$		$\beta_{VP} = 0.015876$
		$\beta_V = 1.0$		
		$\beta_{NP} = 0.18$		$\beta_{NP} = 0.01296$
			$\beta_P = 1.0$	$\beta_{PP} = 0.18$
				$\beta_{NP} = 0.18$
astronomers	saw	stars	with	ears

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 4      11 / 14

Again, so, if you keep on doing that this is what you will see. So, we had filled in this entry and this entry and this entry, but I will encourage you to fill in all the entries of this table and see if you can get to this particular term what is the probability of the sentence generating this whole thing. So, now, if you see this table you also get the answer for the second question is how do I find out the probability of my sentence use the same probabilities and compute beta as for 1 to m or 1 to 5 in this case that will give you the probability of this whole sentence. So, this was about inside probabilities.

Now, how do I compute the outside probabilities? So, if you remember, this is analogous to what we did in the case of forward backward algorithm. So, now, inside probabilities, we are computing bottom up and this will be computed top down and what will be the base case? Base case would be very very simple that is can I generate the whole sequence 1 to m and now because this is the grammatical sentence we are assuming. So, only the first non terminal that is sentence or you can write it as N 1 can derive this whole sentence nothing else can derive. So, that is why the base case will be alpha 11 to m is 1 if the so this j is 1 otherwise it will be 0.

(Refer Slide Time: 25:06)

*Outside Probabilities*

Compute top-down (after inside probabilities)

*Base Case*

$$\alpha_1(1, m) = 1$$

$$\alpha_j(1, m) = 0, j \neq 1$$

*Induction*

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 4      12 / 14

If I write down alpha 1, 1 to m will be 1 and alpha j 1 to m will be 0, if j is not equal to 1, if j is equal to 1 denotes the starting non terminal like s or N 1 whatever we were using the notation form. So, now, this is the base case, now what will be the inductive case. So, inductive case would be I have to compute it in a top down fashion. So, can you think of inductive case here?

Now what I am showing here what are the 2 possibilities of this inductive case. So, let me just show it quickly on the paper.

(Refer Slide Time: 25:54)

© CET  
I.I.T. KGP

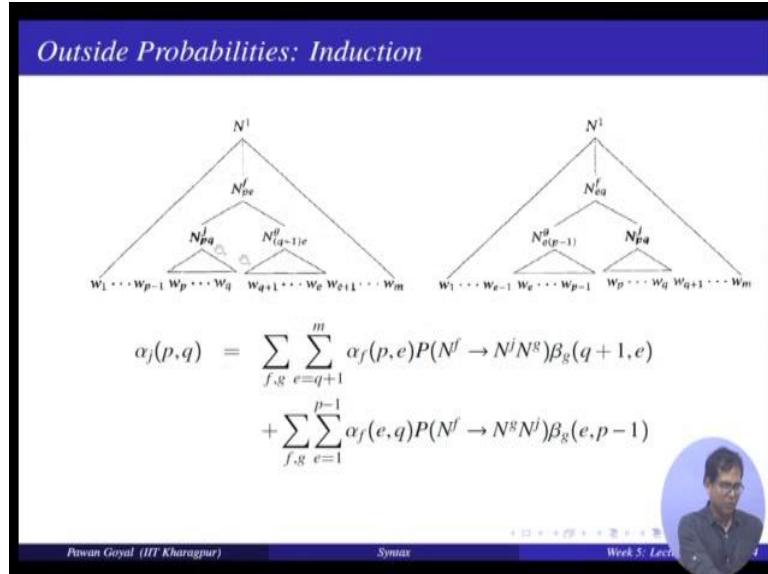
$$\begin{aligned}
 &= p(w_1 \dots w_{p-1} | N^f_{pe}) \cdot p(N^f_{pe} | N^f_{pq}, w_{p+1} \dots w_m) \\
 &\quad \times p(N^f_{pq} | w_q) \cdot p(w_q | N^f_{pq}) \\
 &\quad \times p(w_{q+1} \dots w_e | N^g_{(q-1)e}) \cdot p(N^g_{(q-1)e} | w_{e+1} \dots w_m) \\
 &= \sum N^f_{pe} \cdot p(N^f \rightarrow N^f_{pe}) \cdot p(N^f_{pe} | N^f_{pq}, w_{p+1} \dots w_m) \\
 &\quad \times p(N^f_{pq} | w_q) \cdot p(w_q | N^f_{pq}) \\
 &\quad \times p(w_{q+1} \dots w_e | N^g_{(q-1)e}) \cdot p(N^g_{(q-1)e} | w_{e+1} \dots w_m)
 \end{aligned}$$

I have to compute  $\alpha_{j,p,q}$  and I am going top down, now what is the  $\alpha_{j,p,q}$ ? I have a non terminal  $N_j$  that derives  $W_p$  to  $W_q$  and everything else  $W_1$  to  $W_p - 1$  and  $W_q + 1$  to  $W_m$  is there in the sentence. Now I have to do it in a top down manner. So, I have to use the probability from the upper label. So, I would say what is the non terminal that derives this  $N_j$  with some other non terminal and again this can be either left child or right child. Let us take the case where it is the left child. So, I would say there is some non terminal  $N_f$ , it is starting from  $P$  up to you have to take some  $W_e P_e$  that derives  $N_j$  and some  $N_g$  and what is the parameters here this is  $p, q$  this would be  $q + 1$  to  $e$ . So, this is deriving  $q + 1$  to  $e$ .

Now, I am doing it top down. So, I have already completed this that is the probabilities of this sequence this and  $W_e + 1$  up to  $m$ , this I have already obtained. Now what I have to obtain in  $\alpha_{j,p,q}$  is the probability of  $W_1$  to  $W_p - p N_j p q$  and  $W_q + 1$  to  $W_m$ , this is what I have to obtain. Now here I have already obtained this probability and  $N_f p e$  and somewhere  $W_q + 1$  to  $W_m$  this I have obtained. So, how do I write it in this in terms of  $N_f p e$ ? I will say this is  $N_f p e$  although they may be different possible  $N_f p e$ , we will do the summation over that later times probability of this will probability  $N_f$  gives you  $N_j N_g$  already we have come here and this is deriving  $W_q + 1$  to  $W_e$  because I need these probabilities also.

Now, how do I, how do I write down this expression this we just did in the previous case. So, this is nothing but the inside probability. So, this would be  $\beta_{g,q+1,e}$  and that gives me the probability for this particular extraction although there are various summations that we will have to do, same case we can do for the when this is the right child then  $N_g$  will come here and this is a huge very very similar.

(Refer Slide Time: 29:24)



If you look at them in this case so this is where it is a left child, this is where it is a right child, in this case as we did it is alpha f p e probability of this rule and inside probability of this whole sequence; and what are these summing? Over all the possible f and g's because it can be any non terminal that is arriving this N along with any other non terminal; and what is something else that can vary? My e can vary from q plus 1 up to m. So, I am summing over all the possibilities and you can similarly see for the second case.

Now, so what is important here? You can compute this outside probabilities by using inside probabilities on a; that means, you have to first compute the inside probabilities use that to compute the outside probabilities although computing outside probabilities much more difficult than computing inside probability if you want to do that on paper that is why we did a example only for the inside probabilities.

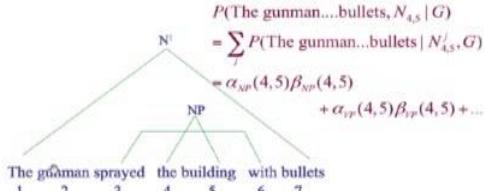
(Refer Slide Time: 30:31)

## Product of inside-outside probabilities

$$\alpha_j(p, q)\beta_j(p, q) = P(w_{1(p-1)}, N_{pq}^j, w_{(q+1)m}|G)P(w_{pq}|N_{pq}^j, G) = P(w_{1m}, N_{pq}^j|G)$$

The probability of the sentence and that there is some consistent spanning from word  $p$  to  $q$  is given by:

$$P(w_{1m}, N_{pq}|G) = \sum \alpha_j(p, q)\beta_j(p, q) = P(N_1 \rightarrow w_{1m}, N_{pq} \rightarrow w_{pq}|G)$$



Pawan Goyal (IIT Kharagpur)

Syntax

Week 5: Lecture 4

14 / 14

$$P(w_{1m}, N_{pq}|G) = \sum \alpha_j(p, q)\beta_j(p, q) = P(N_1 \rightarrow w_{1m}, N_{pq} \rightarrow w_{pq}|G)$$

Now, coming to this point so, remember when I were saying that we have done this parameterization of inside and outside probabilities such that if we multiply that we had something interesting that we can use further and the goal here is also to compute the rule probabilities that is my third question. I was asking that how do I compute my rule probabilities of PCFG? We will come to that in the next lecture.

Let us quickly see what do I get if I multiply the alpha  $j$   $p$   $q$  with beta  $j$   $p$   $q$ ? So, if I do that so, alpha  $j$   $p$   $g$  is nothing but probability of  $W_1$  to  $W_p$  minus 1  $N_j p q$  and  $W_q$  plus 1 to  $m$  given my grammar and beta  $j$   $p$   $q$  is probability of  $W_p q$  given and  $N_j p q$  in the grammar. So, you multiply both of these, you will see that  $W_p q$  can come here and  $NP_q$  is also retained here because here I need  $N_j p q$  to be given that is already given to me before. So, this is the multiplication, now what does this say? So, it is saying if I multiply alpha  $j$   $p$   $q$  with beta  $j$   $p$   $q$ , I get the probability of generating this whole sequence of words  $W_1$  to  $W_m$  and that a non terminal  $N_j$  derives  $W_p$  to  $W_q$  given my grammar.

Now, suppose I want to find out the probability that any possible non terminal derives from  $W_p$  to  $W_q$ , there is some consistent spanning from  $W_p$   $W_q$ , in that case, I will have to sum over all possible  $j$ s that I can achieve by summing over all possible  $j$ s for alpha  $j$   $p$   $q$  into beta  $j$   $p$   $q$ . This is the probability of the sentence and that there is some consistent is spanning from the  $W$  word  $p$  to word  $q$  and this is you can write in this manner this is the same thing  $N_1$

derives W 1 m and NP q derives W p q this and this are the same thing except this particular index of N j.

Now, if we go back to the previous example that we took what does that mean. So, probability is that this whole sentence; the gunman sprayed the building with bullets and that there is some consistent spanning from word 4 to word 5; that means, there is some non terminal, they derive this sequence the word 4 to word 5, how do I obtain it? This should be summation over all the possible non terminals N j such that this happens the gunman sprayed the building with bullets N j p q given g and this is nothing, but alpha NP 4 5 beta NP 4 5. So, here I am taking all possible j's, j can be NP V p and so on.

Whatever non terminals so I multiply alpha beta for a particular j and add overall the possible j that gives me this probabilities, so, even here it may not be very very clear what is the actual use of obtaining this particular probability value, how do I use it further and that is what we will see in the next lecture that how do I use this inside outside probability for learning the rule probabilities of my probabilistic context free grammatical. So, I hope this concept of inside outside probabilities is clear to you and then we can see how do use that further in the next class.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 26**  
**Inside-Outside Probabilities**

Welcome back for the final lecture of this week. So, in the last lecture, we had started with the concept of inside outside probabilities and how do you use them for answering certain questions like what is the probability of a sentence as per my grammar and what is the most likely parse. So, I have used the inside algorithm specifically to find out what are the probabilities of this sentence as per my grammar and then in the end I was saying that we will also use this concept to find out the rule probabilities of my grammar and how do we do that again exactly what we will discuss in this lecture.

(Refer Slide Time: 01:00)

*How to get the rule probabilities*

*Parsed Training Data*

You can count!

$$\hat{P}(N^j \rightarrow \delta) = \frac{C(N^j \rightarrow \delta)}{\sum_{\gamma} C(N^j \rightarrow \gamma)}$$

*But what if the training data is not available?*

i.e. gold standard parse is not known.

- Underlying CFG is known and we are given a set of sentences
- For each sentence, we can find out all the possible parses
- Maximize the likelihood of the sentences in the data under the PCFG constraints

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lec

$$\hat{P}(N^j \rightarrow \delta) = \frac{C(N^j \rightarrow \delta)}{\sum_{\gamma} C(N^j \rightarrow \gamma)}$$

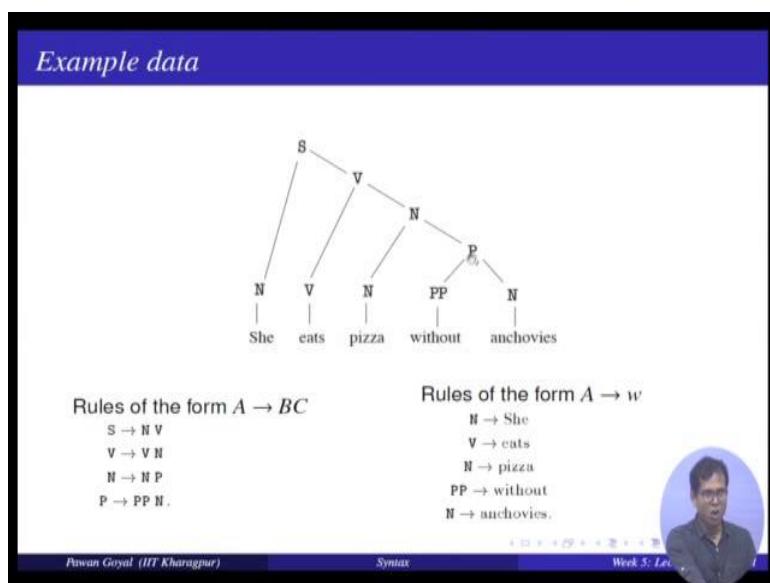
Now in general, how do you obtain the rule probabilities and remember this is very similar to what we were also talking about in the case of HMMs, when I have to learn the parameters of my HMM, I can do that in 2 manner, one where I am given already the labeled data set; that means, I am given what are the sentences and what is their past reach. If I am given all that

so, then from there I can compute how many times a particular non terminal derives a particular sequence divide by number of times a particular non terminal has been used that will give me the probability of this particular rule. If there is only one rule possible from  $N_j$  this is always you want, but if there multiple rules are possible I can find out what fraction of times this particular instance has been used in my labeled data sets and by that I can compute all the rule probabilities.

Now, so this is easy, but what about the case where the training data is not given to us that is I have no way to find out in which sentence what is the rule that has been used. So, what is in fact, given to us? So, we are; assume that we are given the underlying context free grammar. So, I am given all the possible rules, but what I am not given what is the probabilities for each individual rules so; that means, the particular PCGFs part the rule probabilities is not given to us and I am given a lot of sentences and I am the given the grammar that will generate these sentences, but not the rule probabilities and my task is how do I find out these rules probabilities.

So, in other words how do I find out the parameters of my PCGFs and we will use the same sort of idea that we did earlier that is you are given the observation of sentences find out the parameters of a model that maximize the likelihood of observing the sentences. So, this is what we are going to do maximize the likelihood of the sentences in the data under the PCGF constraints and for that we will use some sort of expectation expression algorithm.

(Refer Slide Time: 03:25)



Let us take a simple example and what is the intuition for using this. So, we start with this simple sentence like she eats pizza without anchovies and a particular parse; tree is given to us. So, what kind of rules do you see here? So, you have rules of the form a single non terminal derives to 2 different non terminals like S derives N and V, V derives V and N and so on and there are rules of the form a non terminal derives a terminal like here and derives anchovies PP derives without N derives pizza V derives eats and so on. So, these are different sort of non terminals that are given the rules that are given to me I do not know the probabilities for each individual rule.

Now, can you think of any other parse for this sentence, she eats pizza without anchovies, so one parse that we saw was she eats pizza and pizza without anchovies modifying pizza what is the other possibilities. So, other could be the without anchovies does not modify pizza it is a separate phrase altogether starting from V, so something like she eats pizza with fog.

(Refer Slide Time: 05:00)

*Example data*

Is any other parse possible for *She eats pizza without anchovies* syntactically?  
Consider *She eats pizza without hesitation*

```

graph TD
    S --- N1[She]
    S --- V1[V]
    V1 --- eats[eats]
    V1 --- pizza[pizza]
    pizza --- N2[pizza]
    pizza --- PP1[PP]
    PP1 --- without[without]
    PP1 --- N3[hesitation]
  
```

*New Context-free rules:*

$V \rightarrow V \ N \ P$   
 $N \rightarrow \text{hesitation}$ .

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lec 1

In that sort of meaning so yeah so, she eats pizza without hesitation that is another sort of possibility, now a single V here is giving me V and N P. So, this is the different parse tree for the same sentence

Now, I can what are the new rules that I have added V gives me V N and P and N gives me hesitation these are 2 new rules I have added. So, now, what do you see I have 2 different sentences; she eats pizza without anchovies and she eats pizza without hesitation and both

have 2 possible parse edge and I know what are the all possible rules now my task is given these sentences how do I find out what should be the ideal rule probabilities.

(Refer Slide Time: 05:49)

## Estimating the model parameters

$$\sum_{\alpha} \Phi(A \rightarrow \alpha) = 1$$

Now for that let me first define so we need to find out all these probabilities. So, probability of this rule S gives me N and V probability of rule N gives me pizza and so on, now what is the PCGF requirement? The PCGF requirement is that it starting from a single non terminal all the possible sets of right hand side that I can generate (Refer Time: 06:11). So, that is so, all the possible rules starting from a to any alpha the probability should also added to 1 so now, if I look at my grammar so, I have 5 rules pair and occurs in the left hand side N gives me NP, N gives me pizza anchovies and so on. So, all these should also add to 1.

Similarly, 3 rules where V occurs in the left hand side so, all these should also added to 1 and then there are some other rules like S gives me N V, N V nothing else, this would be 1 and so on. So, this constraints I can obtain from my grammar, I know what are the rules and what is the constraint starting from the left hand side all possible rule should have a probability adding up to 1.

(Refer Slide Time: 06:58)

*Likelihood computation*

$W_1 = \text{"She eats pizza without anchovies"}$

$W_2 = \text{"She eats pizza without hesitation".}$

$P_\phi(W_1, T_1) = \phi(S \rightarrow N\ V) \phi(V \rightarrow V\ N) \phi(N \rightarrow N\ P) \times$   
 $\times \phi(P \rightarrow PP\ N) \phi(N \rightarrow She) \phi(V \rightarrow eats) \times$   
 $\times \phi(N \rightarrow pizza) \phi(PP \rightarrow without) \phi(N \rightarrow anchovies)$

$P_\phi(W_2, T_1) = \phi(S \rightarrow N\ V) \phi(V \rightarrow V\ N\ P) \phi(P \rightarrow P\ PP) \times$   
 $\times \phi(N \rightarrow She) \phi(V \rightarrow eats) \phi(N \rightarrow pizza) \times$   
 $\times \phi(PP \rightarrow without) \phi(N \rightarrow hesitation)$

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 1

Now, I have 2 sentences, can I compute the likelihood of the sentences W 1 and W 2. So, what do I mean by likelihood what is the probability of generating the sentence as of my grammar? Right now I am not giving the rule probabilities, but I can write down in terms of the variable rule probabilities. So, what will the likelihood of W 1? So, for that I have to take the 2 possible parse trees. So, here T 1 is the first parse tree.

So, here I am giving the probabilities of both the sentences as per the first parse tree. So, probability of sentence W 1 as per the first parse tree T 1 and probability of sentence W 2 as per the first parse tree T 1, how do I compute that it is very similar to what we saw in the case of PCGF? How do we compute the probability of a parse tree if the rule probabilities are given that was very easy here the possibilities are not given, but you can parameterize. So, you will say what is the probability of S giving me N V and so on up to you go to the leaves and I do not know these whole probabilities similarly I can write down this likelihood of the sentence W 2 as per my first parse tree.

(Refer Slide Time: 08:11)

*Likelihood computation*

$$P_\phi(W_1, T_1) = \phi(S \rightarrow N V) \phi(V \rightarrow V N P) \phi(P \rightarrow P PP) \times \\ \times \phi(N \rightarrow She) \phi(V \rightarrow eats) \phi(N \rightarrow pizza) \times \\ \times \phi(PP \rightarrow without) \phi(N \rightarrow anchovies)$$

$$P_\phi(W_2, T_2) = \phi(S \rightarrow N V) \phi(V \rightarrow V N P) \phi(N \rightarrow N P) \times \\ \times \phi(P \rightarrow PP N) \phi(N \rightarrow She) \phi(V \rightarrow eats) \times \\ \times \phi(N \rightarrow pizza) \phi(PP \rightarrow without) \phi(N \rightarrow hesitation)$$

*Likelihood of the corpus*

Probability of a sentence  $W$ :  $P_\phi(W) = \sum_T P_\phi(W, T)$

If the training data comprises of sentences  $W_1, W_2, \dots, W_N$ , then the likelihood is

$$L(\phi) = P_\phi(W_1)P_\phi(W_2) \cdots P_\phi(W_N)$$

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 5      7 / 11

Similarly, I can do further second parse tree also for both the sentences. So, this tells me. So, if I know all the possible parse trees because my CFGS given to me, I will know all the possible parse tree, I can put my all the rule probabilities as variables and define what is the likelihood of various parse tree now what is the likelihood of the sentence, it will be summation of the likelihood as per different possible parse trees. So, probability sentence such summation over all the possible trees that can generate this  $P \phi W T$ , in this case for both sentences  $W_1 W_2$ , I had 2 different parse trees.

So, I will just add the 2; 2 probabilities to get the likelihood of the sentence and how do I get the likelihood of the whole corpus that has multiple sentences for that I will come to the likelihood of each sentence and multiply those. So, if I have a sentences  $W_1$  to  $W_N$ , I compute the likelihood of each and keep on multiplying now we know that how do I express the likelihood of my corpus in terms of the rule probabilities right the only variable here are all the rule probabilities now I can further define my problem.

(Refer Slide Time: 09:33)

*Likelihood maximization*

**Approach**

Starting at some initial parameters  $\phi$ , re-estimate to obtain new parameters  $\phi'$  for which  $L(\phi') \geq L(\phi)$ . Repeat until convergence

Pawan Goyal (IIT Kharagpur) Syntax Week 5: Lecture 5 8/11

My problem would be, so, this is some sort of E M approach, I will start with some initial parameters phi, phi means the rule probabilities I want to re estimate so that I obtain some new parameters phi prime such that the likelihood of my corpus increases now. So, L phi prime will be greater than equal to L phi and I keep on doing that until I converge. So, now, here we have to apply (Refer Time: 10:04) algorithm so that we can keep on updating our rule probabilities, this is the parameter of my system and how do we do that if you remember that like what we did in the case of learning parameters for G S analogous to that what we will do here we will start with some arbitrary rule probabilities phi and use that to compute something intermediate.

In this case what we will compute? What is the expected number of times a particular rule has been used if the rule probabilities are as per the current parameters? I will compute the expected value; again use the expected value to compute the probabilities. So, I will compute my phi prime again use the phi prime to compute the expected number of times each rule has been used and again compute 5 dwell prime and keep on doing that until you converge and that is why we will be using the inside outside probabilities. So, let us see.

(Refer Slide Time: 11:02)

**Parameter Estimation**

Given some rule probabilities  $\phi$  and training corpus  $W_1, W_2 \dots W_n$ , the new parameters are obtained as:

$$\phi'(\mathbf{A} \rightarrow \mathbf{B} \ \mathbf{C}) = \frac{\text{count}(\mathbf{A} \rightarrow \mathbf{B} \ \mathbf{C})}{\sum_{\alpha} \text{count}(\mathbf{A} \rightarrow \alpha)}$$
$$\phi'(\mathbf{A} \rightarrow w) = \frac{\text{count}(\mathbf{A} \rightarrow w)}{\sum_{\alpha} \text{count}(\mathbf{A} \rightarrow \alpha)}$$

What is  $\text{count}(.)$ ?

$$\text{count}(\mathbf{A} \rightarrow \mathbf{B} \ \mathbf{C}) = \sum_{i=1}^N c_{\phi}(\mathbf{A} \rightarrow \mathbf{B} \ \mathbf{C}, W_i)$$
$$\text{count}(\mathbf{A} \rightarrow w) = \sum_{i=1}^N c_{\phi}(\mathbf{A} \rightarrow w, W_i)$$

$c_{\phi}(A \rightarrow \alpha, W_i)$  is the expected number of times  $(A \rightarrow \alpha)$  is used in generating the sentence  $W_i$ , when the rule probabilities are given by  $\phi$ .

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 5      9 / 11

Idea is that I start with some rules probabilities phi and I am given a corpus that that what are sentences that I observing  $W_i$   $W_j$  and I will obtain the new parameters phi prime using the simple idea. So, this is something that we were saying if we are given the labeled data that is why I will compute the rule probabilities. So, I am saying I can always define probability of the rule  $A$  given  $B \ C$  as the number of times the rule  $A$  given  $B \ C$   $A$  gives goes to  $B \ C$  or a derives to  $B \ C$  has been applied in my corpus divide by all the possible all the different times where  $A$  derives alpha for all possible alpha and this gives me the probability for a deriving  $B \ C$ .

Similarly, I can compute probability  $A$  deriving  $W$  by saying how many times, this rule has been used divide by number of times  $A$  gives me alpha has been used in my corpus in my actual corpus, but we do not have any labeled corpus, we only know what parse are possible and for each parse, we can compute the probabilities using the previous parameter phi. So, how do I write down this count  $A$  derives  $B \ C$  number of times, this rule has been used for that I use the idea that I have multiple sentences any sentence I can find the expected number of times this rule  $A$  deriving  $B \ C$  has been used. So, that is count  $A$  deriving  $B \ C$  is nothing but summation over all, the sentences number of times  $A$  deriving  $B \ C$  has been used for the particular sentence same 1 for the count of a deriving  $W$  each sentence find out the expected number of times a particular rule has been used, now how do I actually come up with this formulation expected number of times, a particular rule has been used in a sentence and for that we use the inside probabilities and outside probabilities.

(Refer Slide Time: 13:17)

*Computing Expected counts*

*Inside probabilities*  
The nonterminal  $A$  derives the string of words  $w_i, \dots, w_j$  in the sentence :  
 $\beta_{ij}(A) = P_\phi(A \Rightarrow^* w_i \dots w_j)$

*Outside probabilities*  
Beginning with the start symbol  $S$  we can derive the string  
 $w_1 \dots w_{i-1} Aw_{j+1} \dots w_n : \alpha_{ij}(A) = P_\phi(S \Rightarrow^* w_1 \dots w_{i-1} Aw_{j+1} \dots w_n)$

*Expected count*

$$c_\phi(A \rightarrow BC, W) = \frac{\phi(A \rightarrow BC)}{P_\phi(W)} \sum_{1 \leq i \leq j \leq k \leq n} \alpha_{ik}(A) \beta_{ij}(B) \beta_{j+1,k}(C)$$

$$c_\phi(A \rightarrow w, W) = \frac{\phi(A \rightarrow w)}{P_\phi(W)} \sum_{1 \leq i \leq n} \alpha_{ii}(A)$$

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 5      10 / 11

Now coming back to this inside and outside probabilities and how do we use that to compute the expected number of times a rule has been used in a sentence now this must be clear by the previous slide that if I can compute the expected number of times, a rule has been used in a sentence, I can keep on updating my parameters, this is the only bottleneck, in the previous computation and we will see how to do that using the inside and outside probabilities. So, let me give the definitions again. So, inside probability is starting from non terminal  $A$ , I derive the words  $W_i$  to  $W_j$  in the sentence so that is probability that  $A$  derives  $W_i$  to  $W_j$  as per my grammar.

And the outside probabilities starting from the symbol  $S$ , I can derive the string  $W_1$  to  $W_i$  minus 1  $A$  and  $W_j$  plus 1 to  $W_N$ . So, it starting from  $S$ , I can derive  $W_1$  to  $W_i$  minus 1  $A$   $W_j$  plus 1 to  $W_N$ , now once we are given the inside and outside probabilities, we can actually compute the expected number of times, the rule has been used and the expression comes out to be this one expected number of times a rule  $A$  has been  $A$  derives  $B C$  has been used in my sentence  $W$  each the rule probability  $a$  given  $B C$  divide by the probability of the sentence and this very peculiar term that you see that you are seeing here. So, you are seeing here  $\alpha_{ik} A \beta_{ij} B$  and  $\beta_{j+1,k} C$ , now how do I actually come up with a term like that and how do I come up with this expression that is expected count is given by this. So, for that let us go back to what we were discussing in the last lecture that I can multiply inside and outside probabilities to know something about the probability for the sentence.

(Refer Slide Time: 15:16)

The image shows a handwritten derivation of a formula. At the top, it starts with  $\alpha_j(P, q) \beta_j(P, q) = P(N^j \Rightarrow w_{1m}, N^j \Rightarrow w_{pq} | G)$ . This is followed by an equals sign and a fraction where the numerator is  $P(N^j \Rightarrow w_{1m} | G) \cdot P(N^j \Rightarrow w_{pq} | N^j \Rightarrow w_{1m}, G)$  and the denominator is  $P_\phi(W)$ . To the left of the fraction, there is a note "Exp. # times  $(N^j \Rightarrow w_{pq})$  is used". Below the fraction, there is another note "Exp. # times  $(N^j \Rightarrow N^r N^s)$  is used". The denominator  $P_\phi(W)$  is then expanded as  $\sum_{p=1}^M \sum_{q=p}^m \frac{\alpha_j(P, q) \beta_j(P, q)}{P_\phi(N)}$ . At the bottom, there is a tree diagram showing the derivation of  $w_p - w_q - w_r - w_d - w_{d+1} - w_e$  from  $N^j$ .

Let us go back to that. So, what we were saying if I multiply alpha j p q and beta j p q what does it give me? It gives me the probability that it starting from N 1, I can derive W 1 m and it starting from N j, I can derive W p q as per my grammar. So, now, I can use the chain rule here to write it like that. So, its probability N 1 derives W 1 m. So, this means it derives in any number of steps given by grammar times probability N j derives W p q given N 1 derives W 1 m and my grammar. So, now, what is this probability that N one derives W 1 m given by grammar second write as the P phi W probability of the sentence and what is this say what is the probability that this rule N j has been used to derive W Pp q given that the sentences there and my grammar is there.

And how many times this has been used in this particular context? Only 1 time, so, can I write down expected number of times N j derives W p q is used that would be alpha j p q beta j p q given divided by P phi W and suppose because I do not want to fix this p q, I just want to say expected number of times the rule N j has been used. So, each time it has been used only once for deriving W to W q. So, here I will have to sum over all the possible p and q. So, I will say p can go from one to m suppose there are W 1 to W m. So, these are number of words and q can go from p up to m. So, this is the expected number of times my rule and j has been used.

Now, what is something that I have to express I want to find out for example, expected number of times a rule like A goes to B C has been used or a goes to W has been used what is

the expected number of times these has been used now for that suppose let us take the easy case expected number of times the rule a goes to W or a derives W has been used. So, in this case what I am saying N j derives a particular terminal here that is some p p. So, I can write the beta j p p for that case and beta j p p is simply the rule probability that is what is the probability that in non terminal derives this word W p, so we will see the expression for that so, this one is easy.

But what about this case when the rule A derives B C so, in the particular notation that I have written suppose you want to say expected number of times N j derives N r, N S is used . So, now, what would happen? So, this beta j p q is when the terminal N j derives the whole sequence p q, W p to W q and can use any possible rules yes N r N S or N j N z whatever it can use any non terminals now what I am my limiting I am saying this rule should only use this. So, this non terminal should only derive N r N s. So, then I am saying so; that means, my N j will derive N r N s and this N r N s will again derive say W p to some W d and this will derive W d plus 1 to W q.

Now how do I modify this equation? So, alpha j p q is the outside probability that will remain the same nothing has changed for outside probability, but inside probability because I am saying this should be the situation. So, I further express it like with a particular path. So, I will write in place of beta j p q, I will write probability of the rule N j derives N r N s times this beta probability that is beta r p d times beta s d plus 1 to q, but now the d can vary, I have already been given that N j gives me N r N s, but they can take different possible d s. So, this will be summation over d and d can vary from p to q minus 1. So, these between p and q now if you put that can you see that you can you can actually obtain the same expression that was given in the slide.

If you go back to the slide that is what we have been doing here. So, you see the expression we have 3 parameters beta I j b. So, 3 parameters i j and k that corresponds to p d and q and this was the outside probability and this is the inside probability for the two children; i j and j plus 1 k and then you have the rule probability here. So, this expected number of count has been derived in this particular form and same thing you can try with the next formula the expected number of times the rule a deriving the W has been used in my graph and you will obtain this particular expression.

What we are seeing here suppose I start with some initial rule probabilities. So, I can use the inside outside probability. So, all the recursive formulation to compute all the inside outside probabilities for my various rules and stages once I do that I can compute what is the expected number of times each and each individual rule has been used in my corpus as per the current parameters. Once I have the expected number of times rule has been used I can further estimate my parameters by number of times the rule used divide by number of times any particular rule starting with that non terminal has been used and that will give me the new parameters, again I will compute inside outside probabilities expected count, re-estimate the parameters and this I will continue until this converges.

(Refer Slide Time: 23:47)

*And how to compute inside-outside probabilities*

Inductively, as discussed earlier

$$\beta_{ii}(A) = \phi(A \rightarrow w_i)$$

$$\alpha_{1n}(S) = 1$$

Pawan Goyal (IIT Kharagpur)      Syntax      Week 5: Lecture 1

$$\beta_{ii}(A) = \Phi(A \rightarrow w_i)$$

And yeah, so computing inside outside probabilities is as we discussed earlier by this inductive manner. So, what we discussed in this module was that what is parsing in terms of a constituency structure and how do you use the formulation of context free grammar to do parsing, how do we incorporate the rule probabilities there? How do we learn the rule probabilities using this interesting concept of inside outside probabilities? So, I hope by the example that we did in the class, you will be able to understand how it is actual exactly works in practice.

In the next week, we will be starting with this different notion of parsing. So right now, we have done a constituency parsing. So, we will see there is a different notion of parsing called

dependency parsing. So, what is the formulation that that dependency parsing follows? How it is different from this constituency parsing and what are different methods we can use for that that will be a topic for the next week.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 27**  
**Dependency Grammars and Parsing-Introduction**

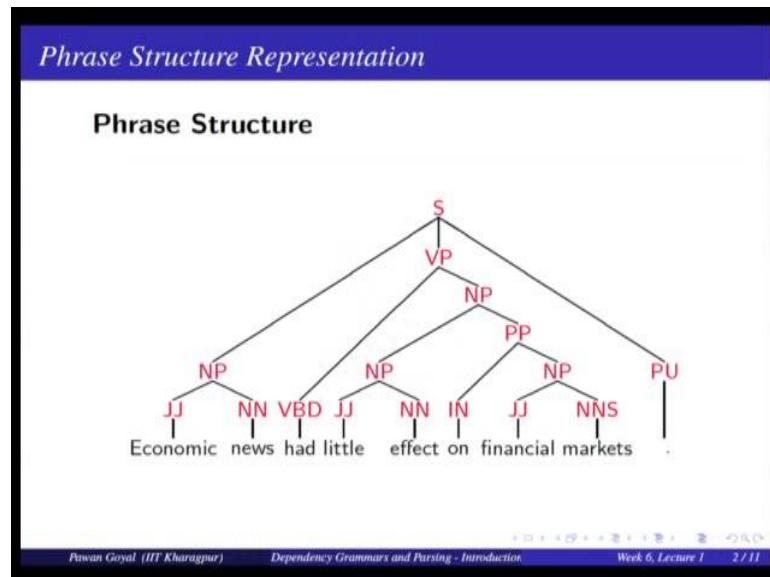
Welcome back for the sixth week of this course. So, we had already started our discussion on what is parsing, so this is the topic in syntax. And we talked about a constituency parsing approach by using the probabilistic context free grammars.

Now, we will use a slight different version of parsing is called dependency parsing. So, where instead of what we did in the case of constituency parsing where we were finding out what are the different word groups in terms of noun phrases, adjective phrases, verb phrases, we will directly find out what is the relation between any two words in a sentence. So, dependency parsing notion has picked up a lot in research field in the last few years, but as such it is origin is very very old, many many different linguistic traditions.

For example, for the Sanskrit the famous Sanskrit grammar [FL] that are the dependency relation between verb and different words in the sentences. So for example, if I have a sentence like Ram eats apple a simple sentence I will see the verb eat is the main verb here and what is the relation of Ram, so Ram is subject of this verb and apple becomes an object of this verb. So, like that I am here I am trying to directly say in this sentence if I am in different words are these related somehow and what is the relation between them. So, this is my dependency structure.

In this lecture we will see what is the formal way in which we can define dependency structure, what are different linguistic constraints, etcetera that we need to impose in this structure. And then we will start talking about the some of the models that are very popular for getting the dependency structure it is starting from the sentence.

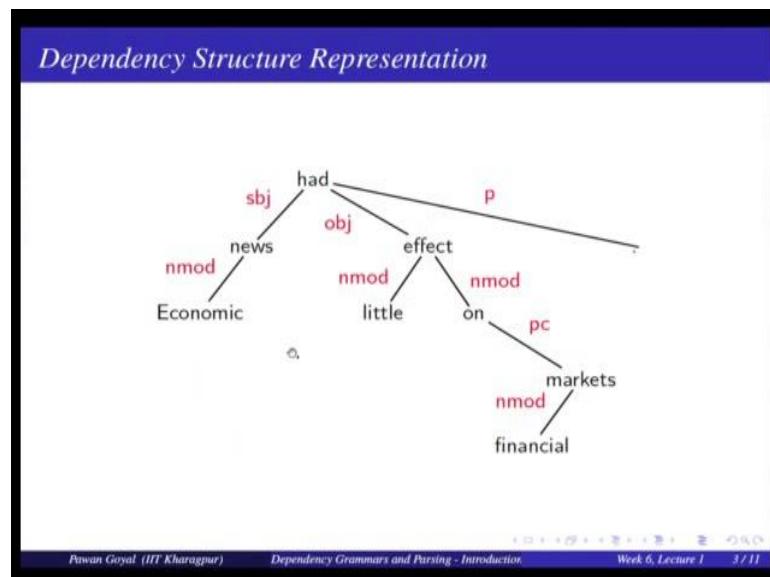
(Refer Slide Time: 02:22)



So, this kind of tree we have seen in the case of phrase structure, also called the constituency structure. So, what we have seen here; I have sentence economic news had little effect on financial markets. So, they are different group of words (Refer Time: 02:36) as single unit; so economic news make a noun phrase.

Similarly this whole thing makes a verb phrase and they together make a sentence. So, like that what we have seen here which words are grouped together, and how this whole sentence has been arranged; that is what you see in a phrase structure.

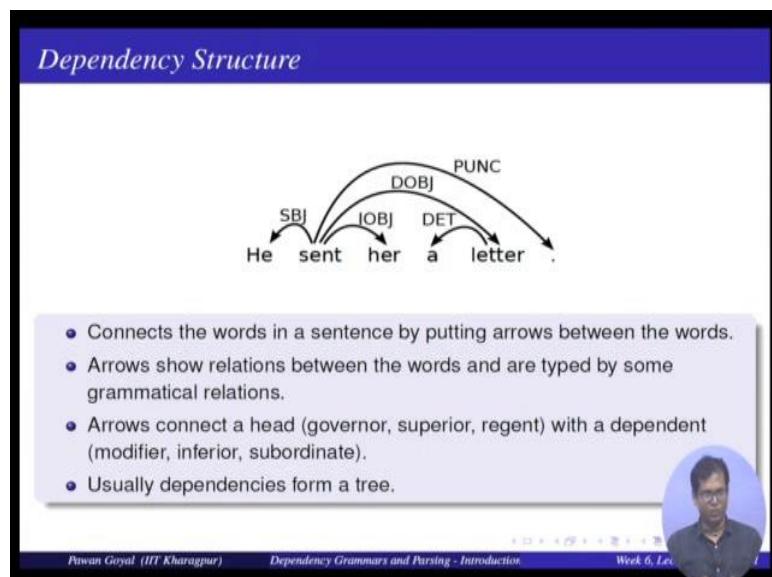
(Refer Slide Time: 03:01)



Now, how dependency structure different. For the same sentence let us look at the dependency structure. Here what are you seeing; now this also looks like a tree, but the nodes here are the words themselves not the phrases. And different words have been connected by some edges and they label by some relation like, had and news are connected and the relation is subject. Saying that news is a subject for the verb had.

Similarly, economic and news are related. So, this is economic is a noun modifier for the word news. And so on lot of different words are connected in this sentence. So, in general what we are seeing two different words in a sentence are connected by certain relation, and this is a directed sort of relation. So, economic and news are related by saying economic is a noun modifier for news.

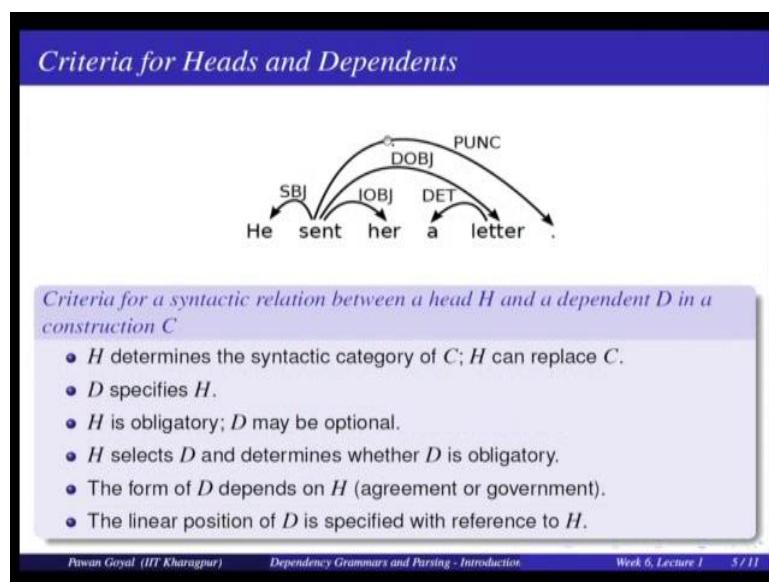
(Refer Slide Time: 03:56)



In general, how we can define a dependency structure? So, we take another sentence here ‘he sent her a letter’. So, what you are seeing here? First you are seeing some 5-6 words or if you take the punctuation. So, they are 5 words as such ‘he send her a letter’ 5 individual words. So, what do I do in dependency structure? So, we are connecting the word in the sentence by putting arrows between the words. So, I am putting an arrow between sent and he and these two are connected. So, ‘he’ is a subject for sent. Similarly I am putting an arrow between sent and her. So, in that ‘her’ is in indirect object for ‘sent’.

Now, what do these arrows show? Which arrows are showing means the relations between words and also typed by some grammatical relation. So, they are saying these some relation and what is the relation, it is the subject relation; 'he' is a subject for 'sent'. And what do these arrows connect? They connect a head word like sent with a dependent word like he. And there are some in linguistic there are some other terms like head can be called governor superior regent and the dependent can also be called a modifier, inferior or subordinate. Usually, as you would see the dependency which will form a sort of tree structure.

(Refer Slide Time: 05:19)



So, what is important here? So, dependencies you have a head that connects to a dependent by an arrow, so arrow points from a head to a dependent. So now, are they said some formal criteria for saying in a sentence if two words are connected what will be a head and what will be the dependent. So, there are some (Refer Time: 05:39) they may not be applying everywhere, but may be applying it certain points. So, you will see some say examples for some of these.

So, what are these criteria for? Defining a relation between head and dependent in a construction; so first criteria you can see is that the head determines what is the syntactic category of the dependent or sorry, what is syntactic category of the whole construction. So, like if I see this construction 'a letter' the whole syntactic category governs simply by the word letter. And that is why the head itself; so the word letter becomes the head.

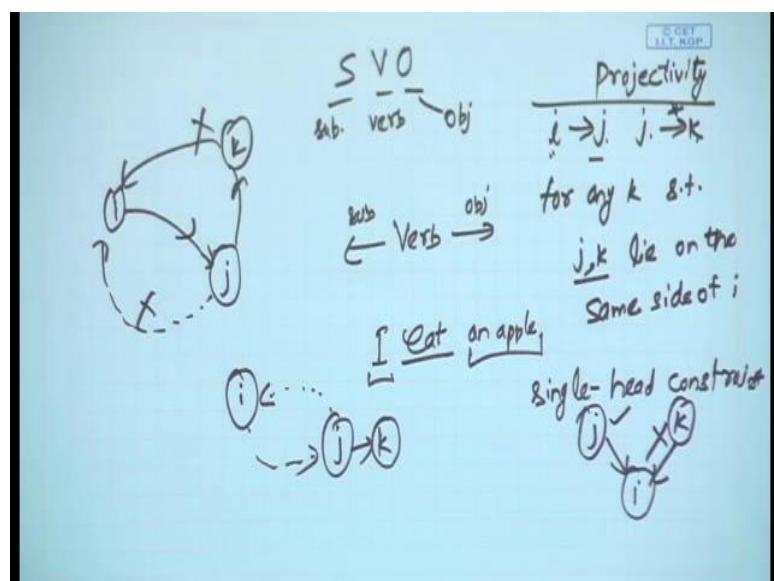
And this word letter can also replace the whole construction C. So, in place of the whole thing a letter I may also write simply letter.

Now D specifies H; that means D is giving further specific information about H. So, if I say only letter may not be as specific as saying a letter. So, all these determine they give some additional information. Similarly here, so sent is the head and letter is the modifier. So, if I just simply say he sent it is not very specific. To whom did he sent, so there I have to specific particular modifier; he sent her. Similarly, what did he sent. So, there I have to put the word letter. So, what you are seeing the dependence is further specifying my head. Now in some cases; so head is always obligatory I need to put head like letter is necessary to put, but D I may put it optional in some cases. So, I may also say he sent her a letter, so D the dependent may be optional in certain cases.

Now, the word the head word it selects my dependent D and it also determines further the dependent is obligatory or not. And in some cases the grammatical form the D takes will also depend on my head. So, this is also called the agreement or government; government in the case of linguistics. So, like the form that use for your verb in your construction will be similar to the form of the subject and object that we are using.

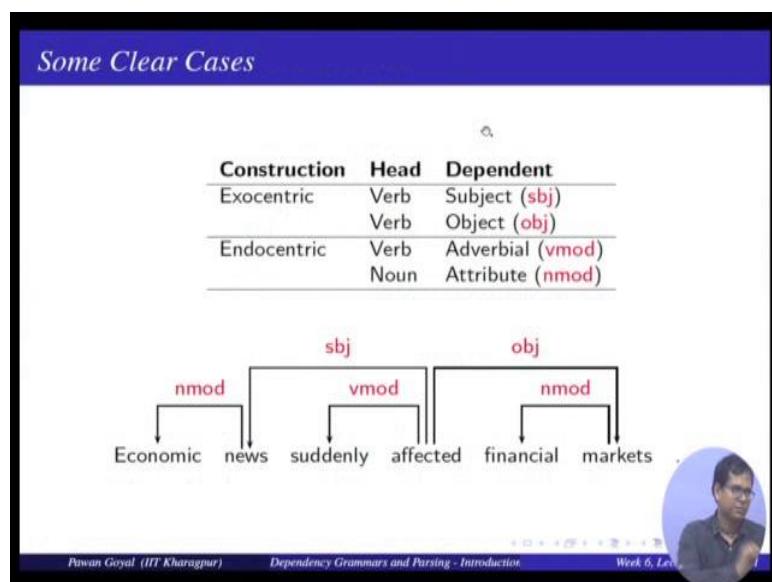
Also with the linear position of the dependent is specified with respect to the head H.

(Refer Slide Time: 08:25)



For example; so English follows this as we are constructed right. So, you get the subject first, then the verb, then the object. So, what we have seen? The linear position of the dependent is determined with respect to the head. So, if I know the verb has come, so I know subject will come to the left and object will come to the right. Like I want to say sentence where I am saying I am eating something. Who is eating the subject will come on the left, I eat and what do I eat say an apple that will come on the right. So, the position of the dependent it is specified with respect to the head, whether it is to the left or to the right.

(Refer Slide Time: 09:09)



Now, there are some cases that are clear where you can easily find the dependencies, so for example if I look at the exocentric and endocentric constructions. What do I mean by endocentric and exocentric? So, endocentric construction is one where, one of the entity here can actually fulfill the whole grammatical function of the complete construction.

So, if you look at this verb and adverbial relation verb modifier, what is the example? Suddenly affected; this may, doing some grammatical function suddenly affected and why it is endocentric because only one word here like affected can fulfill the whole grammatical function for the whole construction suddenly affected. Suddenly is simply modifying that. Same way for non modifier like financial markets, so markets can fulfill the grammatical function for whole thing financial markets. It is not the case for exocentric construction like verb and subject.

So, if I take affect and the news. Affected cannot fulfill the whole grammatical function for news an affected, I need to have the word news there. So, this is called exocentric. Endocentric, one word can fulfill the whole function exocentric a single word cannot. But the cases of exocentric what will become the head and what will become the dependent. Like if I have the verb and subject I know that verb will become the head and subject will become the dependent. Same way verb and object; verb becomes the head and object becomes the dependent.

In the case of endocentric again, the particular word that can fulfill the whole function can become the head and the other word can become the dependent. So, here verb is the head affected and suddenly becomes the dependent. Similarly, here market is the head and financial becomes the dependent.

(Refer Slide Time: 11:11)

**Comparison**

<i>Phrase structures explicitly represent</i>
<ul style="list-style-type: none"> <li>• Phrases (nonterminal nodes)</li> <li>• Structural categories (nonterminal labels)</li> </ul>
<i>Dependency structures explicitly represent</i>
<ul style="list-style-type: none"> <li>• Head-dependent relations (directed arcs)</li> <li>• Functional categories (arc labels)</li> </ul>

Pawan Goyal (IIT Kharagpur)      Dependency Grammars and Parsing - Introduction      Week 6, Lec 1

Now, if you just try to compare what is the difference between a phrase structure a representation and dependency representation. What we see is that in phrase structure we have very explicitly denoting what are the phrases, these non terminal nodes and labels. In the case of dependency relations what are the explicit denoting? They are words, but what are the relation between different words and what is the grammatical category of each individual relation? So, in the case of phrase structure I denote what are phrases and what are the structural categories; like noun phrases, verb phrases, and so on.

In the case of dependency structure what do I represent, what are the various head dependent relations like in the case of directed arcs? And what are the functional categories that are; what are the different arc labels that I am giving to various arcs in my dependent relation?

(Refer Slide Time: 12:07)

**Dependency Graphs**

- A dependency structure can be defined as a directed graph  $G$ , consisting of
  - ▶ a set  $V$  of nodes,
  - ▶ a set  $A$  of arcs (edges),
- Labeled graphs:
  - ▶ Nodes in  $V$  are labeled with word forms (and annotation).
  - ▶ Arcs in  $A$  are labeled with dependency types.
- Notational convention:
  - ▶ Arc  $(w_i, d, w_j)$  links head  $w_i$  to dependent  $w_j$  with label  $d$
  - ▶  $w_i \xrightarrow{d} w_j \Leftrightarrow (w_i, d, w_j) \in A$
  - ▶  $i \rightarrow j \equiv (i, j) \in A$
  - ▶  $i \rightarrow^* j \equiv i = j \vee \exists k : i \rightarrow k, k \rightarrow^* j$

Pawan Goyal (IIT Kharagpur)      Dependency Grammars and Parsing - Introduction      Week 6, Lec 1

Now, formally can I define what is dependency graph? So, I have to define it for the words in my sentence. Formally I can say that the dependency structure can be defined as a directed graph consisting of a set  $V$  of nodes and a set of  $A$  of arcs. Now this is a very generic way in which you define a graph. In a graph you have a set of nodes and set of edges that are connected different nodes. Now is there something more than this (Refer Time: 12:40) in the case of dependency structure.

Now in the case of dependency structure; so this is the labeled graphs, what are the nodes in my graph? The nodes in the graph denote the word forms, the word that I am counting in my sentence. And with that I might also have some annotations like what is the part of speech category of these words. (Refer Time: 13:04) I can also put in my nodes. And what are the edges they are labeled edges with the dependence relations. What is the dependency time if it is subject, object, noun, modifier, verb, modifier etcetera?

And some simple notations that we used for this is that if I am saying that word  $w_i$  is connected to word  $w_j$  with dependency relation  $d$ . I can use this arc  $w_i, d, w_j$ ; it says

that  $w_i$  is the head,  $w_j$  is the dependent and  $d$  is the relation. So, I can say eat - subject or he there is another way in which I can note the arcs by saying this is the headword this is the dependent word I am denoting an arc from a head to dependent word and putting a label on top of that this is another way. And this is equivalent to saying  $w_i, d, w_j$  are in the set of arcs.

There are some other conditions like; if I say that from  $i$  this relation  $i$  to  $j$  direct relation then I can write that  $i j$  in my set of arcs. So, I am not writing the dependency relation here, but I am saying  $i$  and  $j$  are in the set of my dependency relations. Similarly, I can do a closure of this saying that from  $i$  there is a path to  $j$ ; that means in any number of steps I can go from  $i$  to  $j$ . If and only if either  $i$  and  $j$ 's are equal or there is some case such that I go from  $i$  to  $k$  and from  $k$  you can go to  $j$  any number of cases. This is very standard way in which you define the closure relations.

So, with the dependency labels I can define a single dependency or I can also define a closure of that. So, I hope this notation is clear. Once we have this notation what does the total form of conditions that we can put on the dependency graph. On the graphs so that the denoted dependency graph for a sentence, what are the formal conditions?

(Refer Slide Time: 15:24)

### Formal conditions on Dependency Graphs

- $G$  is connected:
  - For every node  $i$  there is a node  $j$  such that  $i \rightarrow j$  or  $j \rightarrow i$ .
- $G$  is acyclic:
  - if  $i \rightarrow j$  then not  $j \rightarrow^* i$ .
- $G$  obeys the single head constraint:
  - if  $i \rightarrow j$  then not  $k \rightarrow j$ , for any  $k \neq i$ .
- $G$  is projective:
  - if  $i \rightarrow j$  then  $j \rightarrow^* k$ , for any  $k$  such that both  $j$  and  $k$  lie on the same side of  $i$ .

Pawan Goyal (IIT Kharagpur)   Dependency Grammars and Parsing - Introduction   Week 6, Lecture 1   9 / 11

So, what are the formal conditions? First condition is that the dependency graph that  $G$  that I get is connected. By connected what do I mean, there is no node in my graph that is isolated from the whole graph. Or in other words if I take any node in my graph I can

always find another nodes such that either that node has an incoming link from that previous node or an outgoing into that node. So, it is connected to some of the node in the graph attached it is not isolated. So, this is the simple condition for saying when the graph is connected.

The second condition is that my graph is acyclic; there is no cycle in the graph. So, what do I mean by that? If I say that from a node  $i$  I have a label, I have a path or I have a dependency from  $i$  to  $j$ . There is no way I can have a path from  $j$  to  $i$ . So, there is no way I can go back to  $i$ , if I am going from  $i$  to  $j$  I cannot go from  $j$  to  $i$  either directly or by following something  $k$  and this, so this is not allowed. So, because this will make a cyclic  $i$  to  $j$ ,  $j$  to  $k$ ,  $k$  back to  $i$ ; so cycles are not allowed in the case of dependency graph.

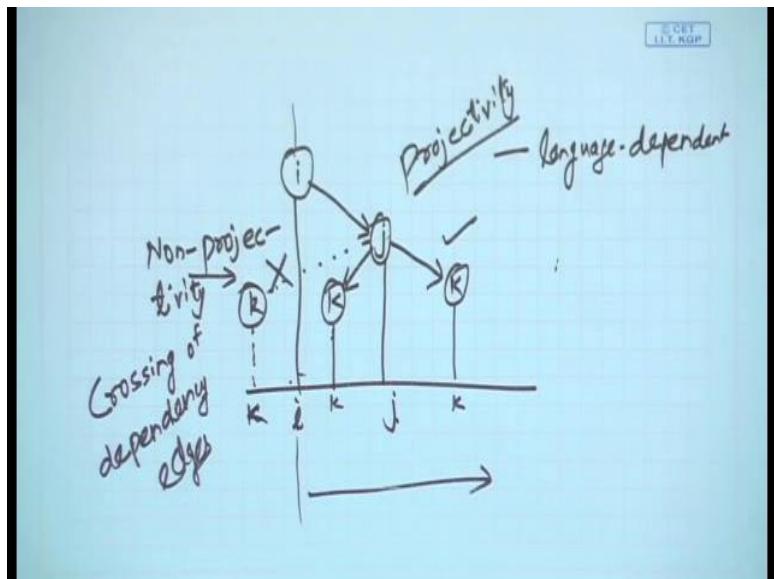
Similarly, if I can show the previous one that means that if I take a node  $i$  in my graph it should not happen that and they are connected and  $i$  is isolated. I can always find some nodes such that either there is an incoming edge or there is an outgoing edge one of these, they cannot be both because of this principle of a (Refer Time: 17:18).

What is the third one? Third condition says that my dependency graph  $g$  obeys the single head constraint. So, what is single head constraint? So, single head constraint means if I take a node  $i$  in my graph they can be at most one head for that. So, if I say that  $j$  is the head for  $i$ ; so  $j$  is the head for  $i$  I cannot find another  $k$  so the  $k$  is also head for  $i$ . This is not possible, there will be only one head for  $i$  node at most is called the Single Head Constraint.

And then finally, we have the fourth constraint that is  $g$  is projective. It says if from  $i$  I can derive  $j$ , and from  $j$  so I am using the word derive here derive means from  $i$ ,  $i$  is the head and I have the dependent  $j$ . Then from  $j$  I have a dependent by some number of edges  $k$ , for any  $k$  such that  $j$   $k$  lie on the same side of  $i$ . And this is the Projectivity Constraint.

And what do we mean by that? If I were edge from  $i$  to  $j$  and from  $j$  I can have in any number of steps I going to  $k$ , then  $j$  and  $k$  should lie on the same side of  $i$ .

(Refer Slide Time: 19:15)



So, that is if I am saying this is my  $i$  and from here I have this relation to node  $j$ . Let me just project them  $i$  is here and  $j$  is here. Then what I am saying is that if  $j$  connects to any dependent either directly or indirectly that  $k$  has to be on the right hand side only. So, there are two possibilities suppose I am taking direct connection I can connect to  $k$  this is allowed or I can connect to a  $k$  here. So, these denote the linear order in the way they are occurring in the sentence.

So,  $i$  and  $j$  are here, so  $k$  can occur either here or here or anywhere else in the right. But, what is not allowed is if  $j$  is related to some  $k$  here that is not allowed, because now  $j$  and  $k$  are on the different side of  $i$ . So, whatever side  $j$  is  $k$  also should be on the same side. So, if this situation happens this is called Non-Projectivity. Also there is another term for that it is called crossing of dependency edges. So, you are seeing here this line and this line are crossing here if I am taking this particular constraint. While they do not cross if I take this constraint or this constraint and you can see if you can go any number of steps they will not cross and this particular constraint is called the Projectivity Constraint.

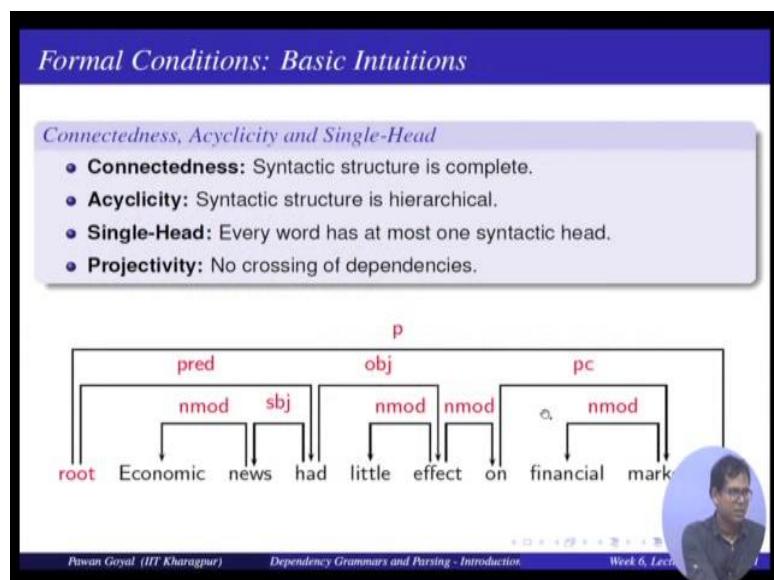
Now, what is important is that this constraint of projectivity is many times dependent on the particular language that you are choosing for your dependency construction. So, it might happen that there are certain languages that do not follow this projectivity constraint and some languages like English they very quite regularly follow this projectivity constraint. So, this is like language dependent. So, the methods that

dependency parsing that we will study in this week, so we will study one method that needs projectivity constraint another method that does not need this constraint.

So, but these four constraints are very very important so that you can come up with the good algorithm for a dependency parsing of the sentence. So, this is an example of what is projectivity. So, if I have a link from a to b I cannot have a link from b to c, because c is on the other side of a.

Similarly if I have a link from a to b I cannot have a link from b to c, because they are crossing.

(Refer Slide Time: 22:23)



In general, what do they mean these four conditions? So, connectedness means that the syntactic structure obtaining my dependency graph is complete. I am not having any isolated nodes that are not connected to my whole structure. In condition of acyclicity means that the structure is hierarchical there is no cycles inside.

What is the single head constraint? It says that every word has at most one syntactic head it cannot have more than one syntactic head. So, in other words we are saying it is determined only by one word not by multiple words. Remember we were saying that the linear position of a dependent sometimes depends on the head. So, node 1 2 different heads to decide the position of the single word and then they might be conflicts the same

word has only one head that determines its syntactic category and in other things. There is no crossing of dependencies that is the last constraint of projectivity that we also saw.

(Refer Slide Time: 23:34)

**Dependency Parsing**

- **Input:** Sentence  $x = w_1, \dots, w_n$
- **Output:** Dependency graph  $G$

**Parsing Methods**

- Deterministic Parsing
- Maximum Spanning Tree Based
- Constraint Propagation Based

Pawan Goyal (IIT Kharagpur)      Dependency Grammars and Parsing - Introduction      Week 6, Lecture 1      11 / 11

So, now once we have these four constraints what is my dependency parsing, so this is how I can define the dependency parsing. I am given a sentence  $x$  that contains words  $w_1$  to  $w_n$  and I want to obtain a output as dependency graph, given this input of sequence of words in my sentence I want to obtain my dependency graph that follows that constraint that we saw earlier. So, it is like single head constraint and projectivity if I am imposing that and all that. So, I want to obtain dependency graph.

So, what are different methods that that we will be using for this? We will talk about different methods so like one method will be deterministic parsing this is a very very popular method. Then a method based on maximum spanning tree. And finally, you can also do it in a constraint propagation method. So, in this course we will talk about the first two methods and how you can use that by having some sort of labeled data and doing some learning from there. So, it is like a data driven parsing. And the final method is something that you will use when if node have any labeled data, but you know what are some of the constraints that your grammars follows. So, can you encode those constraints inside your dependency grammar and obtain dependency graph for a given sentence.

So, this is good for the languages where if node have a labeled data, but you have the labeled data in terms of dependency graphs you probably go for a data driven approach. So, starting from the next lesson we will start talking about these driven approaches for dependency parsing.

Thank you.

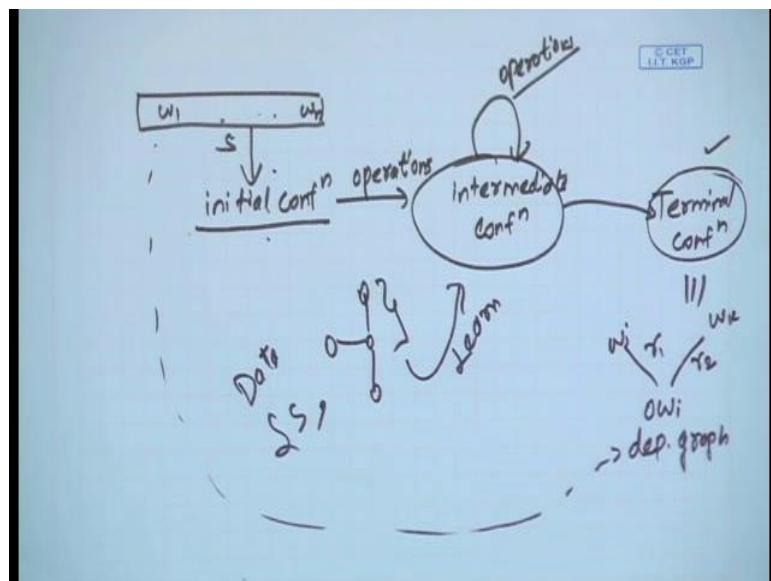
**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 28**  
**Transition Based Parsing: Formulation**

So, welcome to the second lecture of this week. So, we had started for our dependency parsing formulation and we define what is the problem and what are the very formal conditions on a dependency graph in the last lecture. So, we are given input sentence, I want to obtain a dependency graph for that sentence. And further we discussed that there are certain ways we can handle this problem. And in this course, we will mainly focus on approaches that just use some sort of labeled data for this task.

So, it will be some sort of machine learning task for finding out the dependency path for a new sentence. So, now so in the next four lectures in this week, I will cover two different methods for the same problem. But before discussing the methods in detail, let me try to give you a basic intuition on what we are trying to achieve. And this will also help you to understand that if you are given a new problem and you want to take it in a machine, you want to solve it in a machine learning based method, then how you should go about it.

(Refer Slide Time: 01:28)



So, let me take this case. So, I have a sentence that contains some words. This is some arbitrary sentence, this is a new sentence I might not have seen it before. And my goal is to

come up with the dependency graph. The nodes are different words and they have certain relations, r 1, r 2 etcetera are certain relations. This is my dependency graph. So, I want to go from the sentence to my dependency graph. Now, suppose I want to do it in a in some sort of systematic manner, by using some sort of data. So, what I will do. I try to convert that to some sort of configuration format, some initial configuration, so that means, there will be a function that will take any sentence as input and convert that would initial configuration.

Now, I will define some operations that I can make on this initial configuration and that will take me to some intermediate configuration. And I should be able to keep on doing these operations until at some point of time, I arrived at terminal configuration. And terminal configuration should be something that resembles a dependency graph. So, my whole task is now starting from the input sentence convert to a initial configuration apply some well defined operations and go to intermediate configuration until you finally, reach a terminal configuration that may be deterministic, may not be deterministic. So, this is my whole thing. This is my sentence and this is my dependency graph; and operations are something I do not know, but I need to define, I need to find out for a particular configuration, what are the operation that I should take.

Now, where does machine learning comes in here. So, this looks like very deterministic. What is the problem, the problem is they may not be a single operation. So, I should have a way to find out for the given configuration, what is the operation that I should take and that I am trying to learn from the data that I have, so that is where the machine learning comes in. So, we have data where we have a set of sentences and their corresponding dependency graphs. So, there I can find out for what configuration what was the operation I had taken. And I will use that to learn for a new sentence or a new configuration what operation I should be taking, so that given any new sentence I can actually find out its dependency graph. And this is just generic idea we will see specifically how it will be used in the two different methods that we will cover in this course.

(Refer Slide Time: 04:50)

**Deterministic Parsing**

**Basic idea**  
Derive a single syntactic representation (dependency graph) through a deterministic sequence of elementary parsing actions

**Configurations**  
A parser configuration is a triple  $c = (S, B, A)$ , where

- $S$  : a stack  $[ \dots, w_i ]_S$  of partially processed words,
- $B$  : a buffer  $[ w_j, \dots ]_B$  of remaining input words,
- $A$  : a set of labeled arcs  $(w_i, d, w_j)$ .

<b>Stack</b>	<b>Buffer</b>	<b>Arcs</b>
$[\text{sent, her, a}]_S$	$[\text{letter, .}]_B$	$\text{He} \xleftarrow{\text{SBJ}} \text{sent}$

Pawan Goyal (IIT Kharagpur)    Transition Based Parsing: Formulation    Week 6, Lecture 2    2 / 15

So, now the first method that we are going to cover is a transition based parsing method. So, we will see its formulation. So, what is the basic idea? So, it is similar to the intuition that I had provided earlier that is I have to derive a single syntactic representation, it is called the dependency graph via a deterministic sequence of elementary parsing actions and this is what we were saying as operations. I start with the sentence; I keep on having certain operations over that until I arrive at some dependency graph. So, now in this particular case of transition based parsing, what do I mean by a configuration, what do we mean by configuration look like.

So, a configuration, I will need a triple that is S stack, buffer and a set of arcs so that is at any point I can define a configuration. In the configuration, I will have a stack where there will be some set of words that has been partially processed. Then the buffer, it will contain the words that are remaining and whatever arcs I have already obtained via my operations, this will be in the set of arcs. So, at the bottom of this slide, you are seen one example of a configuration. So, this is for the sentence, he sent her a letter. So, this is the intermediate configuration, where you have the words sent, her, a, in the stack; so you are partially processed at this point. The word letter and punctuation are in the buffer they are remaining. And there is already an arc between the words he and sent. And he is the subject of the word sent – the verb sent. So, this is the intermediate configuration.

Now, how do I formally define my whole transition system? So, formally so it will be very analogous to what I started discussing. So, I need to define what are my possible configurations, how do I go from a sentence to an initial configuration and which configuration, I will call as a final configuration. And what are operations I should make so that I can transit from one configuration to another configuration and that is why name of transition system also comes in.

(Refer Slide Time: 07:04)

*Transition System*

A transition system for dependency parsing is a quadruple  $S = (C, T, c_s, C_f)$ , where

- $C$  is a set of configurations,
- $T$  is a set of transitions, such that  $t : C \rightarrow C$ ,
- $c_s$  is an initialization function
- $C_f \subseteq C$  is a set of terminal configurations.

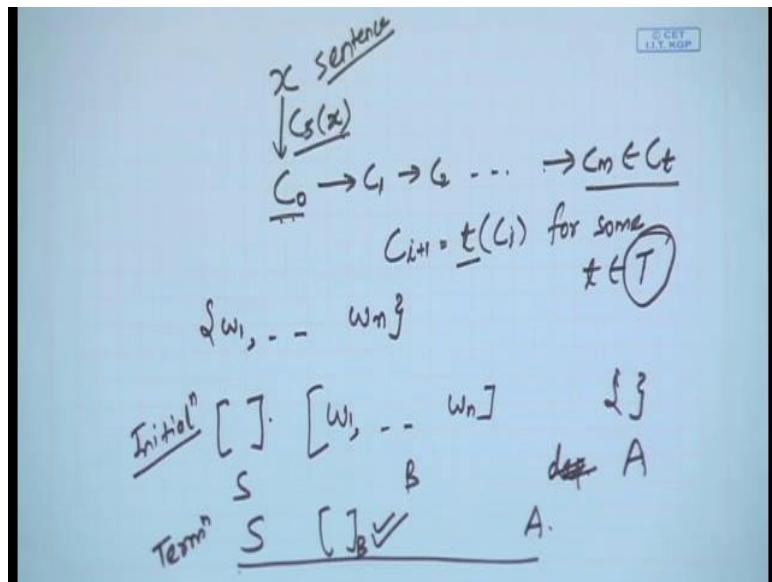
A transition sequence for a sentence  $x$  is a set of configurations  $C_{0,m} = (c_0, c_1, \dots, c_m)$  such that  
 $c_0 = c_s(x)$ ,  $c_m \in C_f$ ,  $c_i = t(c_{i-1})$  for some  $t \in T$

**Initialization:**  $([], [w_1, \dots, w_n]_B, {})$   
**Termination:**  $(S, [], A)$

Pawan Goyal (IIT Kharagpur)      Transition Based Parsing: Formulation      Week 6, Lecture 2      3 / 15

So, then formally we will define dependency parsing transition system as a quadruple that has  $C$ ,  $T$ ,  $c_s$  and  $C_f$ . And what are these, capital  $C$  is a set of configurations, all the configurations that are possible. Now, capital  $T$  denotes a set of transitions and each transition takes me from one configuration to another configuration, from  $C$  to  $C$ ; one configuration in the set  $C$  to another configuration in the set  $C$ . Now,  $c_s$  will be an initialization function that you will give a sentence as input and you will find the initial configuration as output. And then you will have a set of terminal configurations, so; that means, whenever you reach a terminal configuration via your operations, you will stop, anyway call it as your dependency graph.

(Refer Slide Time: 08:17)



Now, whenever you are given a sentence  $x$ , what is my transition sequence. So, a sequence will be a set of configurations, it is starting from  $c_0$  to  $c_m$  and what are the conditions on those  $c_0$  is something that you can derive from the sentence. So, suppose  $x$  is the sentence that you are giving as a input and you want to find out its dependency graph. So, now what is the idea? You have a function – initialization function, you have applied that function over  $x$  and you get a transition sorry you get a configuration  $c_0$ . Now, you keep on applying various operations and get  $c_1, c_2$  up to  $c_m$ ; and  $c_m$  is in the set of terminal configurations. This is my initialization function, set of terminal configurations and  $c_0$  to  $c_m$ , all are in the set of configurations.

And what is the relation between any two conjugative  $c_i$  and  $c_{i+1}$ ? Relation would be I can obtain  $c_{i+1}$  by making a transition over  $c_i$ , for some transition I have defined in my set capital  $T$ . So, this is my process. It starts with the sentence. Apply the initial initialization function get the initial configuration; keep on applying by a various transitions until you obtain a terminal configuration.

Now, as per the transition system that we have defined what will be the initialization. Initialization will be simple. So, I have to define what is my  $S$  stack, what is my buffer, and what are my arcs because when I am starting with the sentence  $w_1$  to  $w_n$ . Initially, my structure contains the set of partially processed words, so there is nothing that is processed right now, so my stack will be empty. My buffer should contain the set of remaining input

words, so remaining words are all the words  $w_1$  to  $w_n$ . And my arcs should contain whatever dependency relation, I have already found, so it will be empty, this will be empty set. So, this is my initialization.

Now, what will be the terminal configuration? A termination, so the idea is I should not have any words remaining in the buffer, so my buffer should be empty. So, terminal condition I can define as stack would have some may or may not some words; buffer should be empty, and I will have obtained a set of arcs. So, this is important, my buffer should be empty. A stack may or may not contain some words, and arcs will have some dependency relations already. So, now once you know what is my initial configuration, what is my termination condition, now what is remaining in this in this whole thing. We know all the possible set of configurations. So, what is remaining is what are all the possible transition that I can take for going from one configuration to another configuration. How can I move from initial configuration by a sequence of operations so that I arrive at a terminal configuration?

(Refer Slide Time: 11:28)

### Transitions for Arc-Eager Parsing

$$\text{Left-Arc}(d) \frac{([ \dots, w_i ]_S, [ w_j, \dots ]_B, A)}{([ \dots ]_S, [ w_j, \dots ]_B, A \cup \{(w_j, d, w_i)\})} \neg \text{HEAD}(w_i)$$

$$\text{Right-Arc}(d) \frac{([ \dots, w_i ]_S, [ w_j, \dots ]_B, A)}{([ \dots, w_i, w_j ]_S, [ \dots ]_B, A \cup \{(w_i, d, w_j)\})}$$

$$\text{Reduce} \frac{([ \dots, w_i ]_S, B, A)}{([ \dots ]_S, B, A)} \text{HEAD}(w_i)$$

$$\text{Shift} \frac{([ \dots ]_S, [ w_i, \dots ]_B, A)}{([ \dots, w_i ]_S, [ \dots ]_B, A)}$$

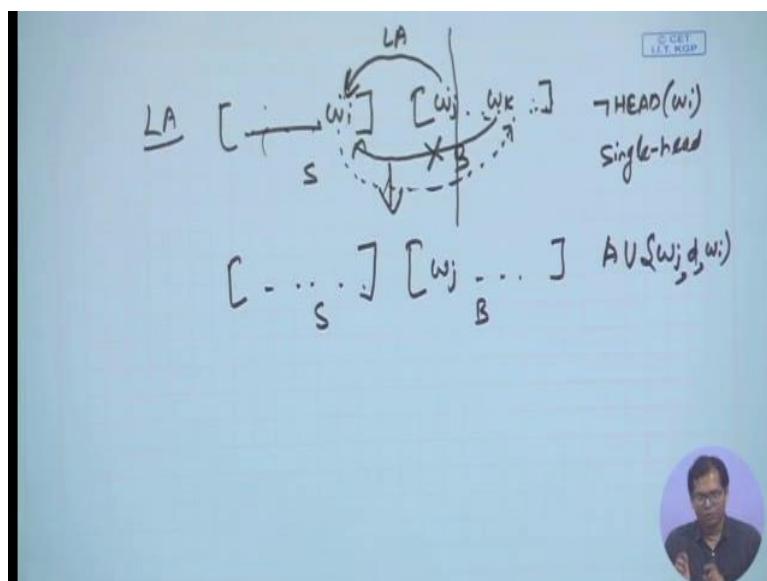
Pawan Goyal (IIT Kharagpur)   Transition Based Parsing: Formulation   Week 6, Lecture 2   4 / 15

So, now in this system, there are four transitions that have been defined, so; that means, for any configuration, you can take one of these four transitions. So, in this slide, I am showing you the four transitions, different transitions that you can take, so how you should read it. So, in the left the names of the transitions, left-arc, right-arc for a dependency  $d$ , reduce and shift, these are the four transitions possible. Now, for each transitions like left-arc, you will have what is the configuration that is the input to that of with which you are starting on top. And

what is the configuration that you will end up with. A starting from this configuration, if you apply a left-arc, what configuration you will arrive at. And finally, in the right, if there is something that is a condition, what is the condition under which you can apply this transition; this is just a necessary condition. You need to have this conditions satisfied to be able to apply this transition.

So, for example, if I see the left-arc transition, what does it say, it says that I can apply it when there is stack contains some words. And I have been shown here the last word in the stack, I can also call it the top of this stack, the word is  $w_i$ . Buffer contains some words  $w_j$ , it is starting from  $w_j$ , top of buffer is  $w_j$ . And I have some set of arcs. Now, what do I mean by left-arc, left-arc means, I making a transition from  $w_j$  to  $w_i$ , yes, this is the left-arc, because the words are occurring in the same order that they occur in the sentence. So, there is a arc from  $w_j$  to  $w_i$ . So, what is the head here,  $w_j$  is the head and  $w_i$  is the dependent. So, now there is also a condition here,  $w_i$  should not already have a head. And let us try to understand this condition.

(Refer Slide Time: 13:53)



So, what am I saying my stack contains some words and top is  $w_i$ ; buffer contains some words starting from  $w_j$ , and I making a left-arc. So, the left-arc as the name says will be from buffer to S stack. And we are we have always operating on the top words in buffer and stack. So, I making an arc from the top word in buffer to the top word in S stack, this is the left-arc. Now, what is the condition? Condition is  $w_i$  should not already have a head.

Now, why is this condition? Just see if I am making this transition, we are saying that the  $w_j$  is the head of  $w_i$ . Now,  $w_i$  already has a head, I cannot apply this, why? Can you see what particular condition in dependency parse, does it violate. So, dependency parse has a single head constraint, so it is a single head constraint, so that says each word can have at most one head. So, if  $w_i$  already has a head, then I cannot apply this particular transition, so that is left-arc can be applied only if  $w_i$  does not already have a head.

Now, suppose if I apply a left-arc, now I start with a configuration like that, what is my real thing configuration, so real thing configuration would be I will remove  $w_i$  from the S stack. It will contain all the words except  $w_i$ . Buffer - will remain as it is. And my arcs will be, I have got a new arc, what is that,  $w_j$  dependent,  $w_i$ . This is my new arc. So, I have removed  $w_i$  from my S stack. Now, you might have this question, why am I removing  $w_i$  from the S stack. If the word is removed, I cannot put it back. So, I can remove a word from the S stack only if I am sure that all its relations have already been captured. Now, the question here is can this word  $w_i$  have anymore relations with any words. So, if  $w_i$  has any relation with the words that occurs before  $w_i$ , it would already have been captured at this point, because these are already partially processed words. So, the only relation  $w_i$  might have is with any words after  $w_j$ .

Now, let us take a scenario, take a word  $w_k$ . Now, what kind of relation  $w_i$  might have with  $w_k$ . It can be either an incoming arc or it can be an outgoing arc. Now, can it can have an incoming arc from  $w_k$ , it cannot; yes, because it can have only one single one head, so this is not possible. Now, what is the other condition, can it have outgoing arc to some word  $w_k$ . Now, what do you think about it, is that possible.

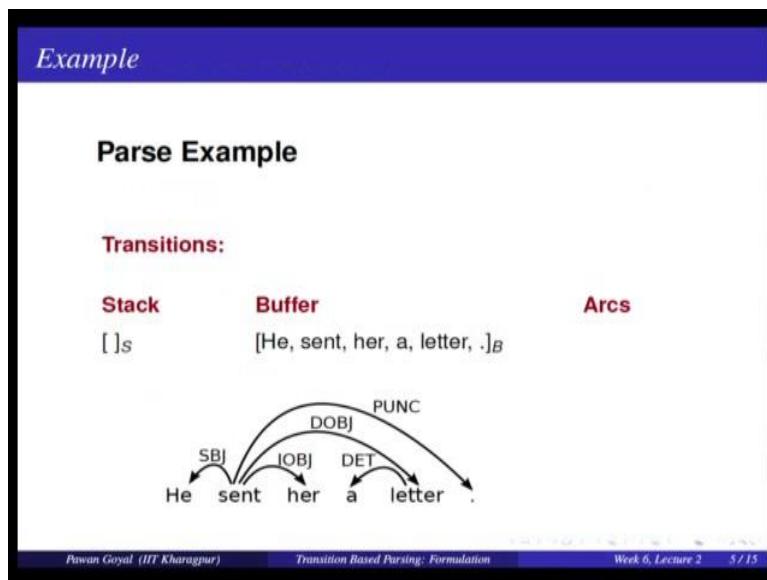
Now, if you think again about the formal conditions that we are defined, what condition is it violating, remember the condition on projectivity. If there is a relation from  $j$  to  $i$  then from  $i$  the next dependent can be only on this side of  $j$ , it cannot be on that side of  $j$ . So, this has to be either in between that is not possible or here. So, this is also not a possibility, so; that means,  $w_i$  cannot have any further relations, so that is why we can remove  $w_i$  from the S stack and this my s stack and this contain everything other than  $w_i$ , buffer remains the same.

Now, let us look at the second transition – right-arc. So, now right-arc is, you are making a relation from  $w_i$  to  $w_j$ ;  $w_i$  is the head and  $w_j$  is the dependent. There is no conditions that you have to see here. And when you apply this particular transition, what do you obtain  $w_j$

goes to the S stack. So, I do not remove W i and W j goes to the S stack and I get a new arc. So, now again you can try to think here why we did not remove W i from the S stack, why we did not remove W j from the buffer and remove it all together, why we are not doing that. And try to think in terms of the constraints that we are put over our dependency parsing. We will come up with some nice intuitions that why we are defining the transitions in this particular manner. So, we have seen the justification for the previous case; now try to justification for yourself for this case. But here the difference is that now the transition is from W i to W j, and accordingly I am getting a new arc.

Then we have two more transitions reduce and shift. What is reduce, in reduce I remove the top word from the S stack and the condition is W i already has a head. If W i already has a head, I can remove that. And finally, the fourth transition is shift, what is that, I take the top word on the buffer and shift it to the top of S stack. And these are the four transitions. Now, the main thing here is when you are at a particular configuration, you might be able to take more than one of these operations or transitions. And your task could be to find out what is the transition that I should take in a particular scenario. So, I hope the four transitions are clear.

(Refer Slide Time: 20:48)



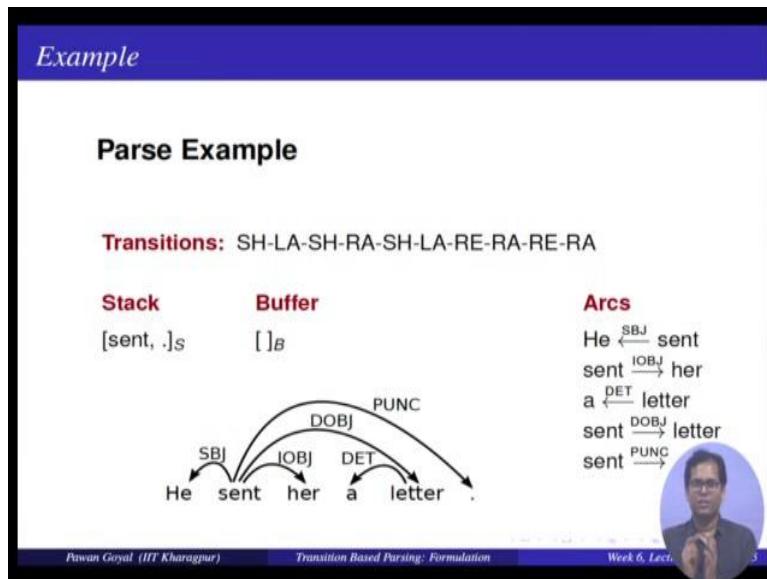
Let us see a particular example and how we do apply these transitions. So, we will take a very simple example. So, what we will have, we will have a sentence – he sent her a letter. And we also know the dependency parse of this sentence. Now, once you are given the sentence

and its actual dependency parse that you want to obtain you have to find out what are the transitions that you should be taking at each intermediate or initial configuration, so that you can end up with the dependency graph. So, let us do this exercise.

So, here this is my sentence – he sent her a letter, and there is also a punctuation mark. How do we start, find out what is the initial configuration. Now, how do we convert this sentence to the initial configuration? S stack would be empty, buffer would contain all the words and arcs would be empty and that is what initial configuration; S stack is empty; buffer contains all the words; and arcs is empty. Now, your task is starts. This is the initial configuration.

Now, what is the transition that you must be taking here? So, this one is relatively easy. I cannot do left arc left arc means say arc from he to word in S stack. There is no word in S stack; they cannot be any dependency relation. Similarly, right arc is not possible because there is no word on the top of s stack; I cannot to reduce because again there is no word on the top of S stack. So, the only transition possible is shift. I can take the top word in the buffer and move that to the stack and this is what I will do. I will take the transition as shift.

(Refer Slide Time: 22:39)



So, if I do right, how do I modify my configuration? So, if I do a shift, he will go from buffer to the S stack and that is what I will obtain now. Now, this is my intermediate configuration. Now, again you defined out what is the transition that he must be taking at this point and that is where you will look at the dependency graph that you want to obtain. And you will see is there any relation you want to establish between the top of s stack and top of buffer.

So, in the parse we are seeing he is a subject for sent; that means, you can establish a relation between sent and he and will be that it should be from sent to he, so it should be a left arc relation so; that means, I can take a left arc transition at this point. And if I take a left arc, what will be the output configuration, I will remove he from the S stack, buffer will remain as it is; and arcs, I will get one more arc and that is what is my next configuration. S stack is empty; buffer contains the remaining words; and I have one arc obtained.

Now, you are at this configuration. Now, what is the transition you should take? Again whenever stack is empty, the only transition possible is shift. So, I will do a shift.

Now, I have one word in the in that stack sent, and there are some words in the buffer her, a and letter. Now, what is the transition you can take at this point? Again you will try to have a look at the dependency parse, you will see if this is relation between sent and her, yes her is a dependent and sent is the head. So, what is this transition, from sent to her, it will be a right-arc. So, I will take a right-arc transition. And what will be the output of the right-arc, the word her should go the s stack and I will get a new relation in my arcs. So, this will be my configuration at this point. I have the words sent and her in the s stack, a and letter in buffer, and I got the arcs.

Now, am I at the point where I have some words in the s stack, I have some words in the buffer. And now I have to see what is the transition I should be taking care. So, let us have a look. Can I take a left-arc or right-arc, you see her and a are not related in my dependency parse, so I cannot take left-arc or right-arc. Now, what are the possibilities, the possibilities are I can either take reduce or shift is both are possible. Now, how do I choose between reduce and shift? And there is a rule of thumb for choosing between rule shift and reduce. And the idea is if you find the (Refer Time: 25:34) words in the stack not on the top, just they were before the top, any word before the top of the s stack that connects to that word on the top of the buffer, you do reduce otherwise shift. Of course, you can do reduce only if you have already found the head for their word.

So, in this case, should I take reduce or shift, let us take this, this rule of thumb. Is there a word in the stack below her that connects to the top of buffer, there is not so I cannot take reduce, I should take shift. So, if I do shift that is what I get. I take a shift; I now have sent her a letter. Now, once I made this configuration, what is the next transition I will take? This

time it is easy, there is a relation between letter and a, this should be left-arc. Left-arc means I removing.

So, now again I am here, there is no relation between her and letter. So, now so; that means, you can choose between reduce and shift, so use the rule of thumb again. Is there a word below the top in the stack that connects to the top of buffer, and you will see, yes. The word sent in the s stack connects to the word letter in the buffer. So, here you can do reduce. And you do reduce and you have the word sent in the s stack, and letter and punctuation in the buffer. Now, is there any relation, yes, there should be a right-arc relation. So, right-arc means letter will go to the s stack, yes. Again, there is no relation between letter and dot – the punctuation. What should I take, is there any relation between any other word in s stack with the buffer, yes, sent and punctuation are related, so I will do a reduce here. Now, sent and punctuation are related by right-arc; and by right-arc, I will take the dot to the s stack. And this is my configuration.

Now, what do you see, should I go any further once I have reach this configuration and you will say no, because what is the condition terminal configuration, the condition is that my buffer should be empty. So, at this point, my buffer is already empty so; that means, I have arrived at a terminal configuration and that is where I will stop. And I have obtained all the possible set of arcs for this sentence. Now, this is an example we have seen just to understand how to take all these transitions. But does that help you to solve this problem whenever you are giving a new sentence, why a new sentence I mean you do not know dependency graph or you might also ask if I already know the dependency parse for a sentence, what is the need for doing all these transitions and that is where the concept of learning will come into picture.

So, we have seen here if we know the dependency graph, we can choose the configuration. At run time, whenever I am given a sentence my whole task is to find out what are the transitions I should be taking at different configurations. And this I will learn from my already labeled data. And this is what we will discuss in detail in the next lecture that is how do I use how do I use concept to define it as a learning problem. And for a new sentence, how do I come up with a dependency graph.

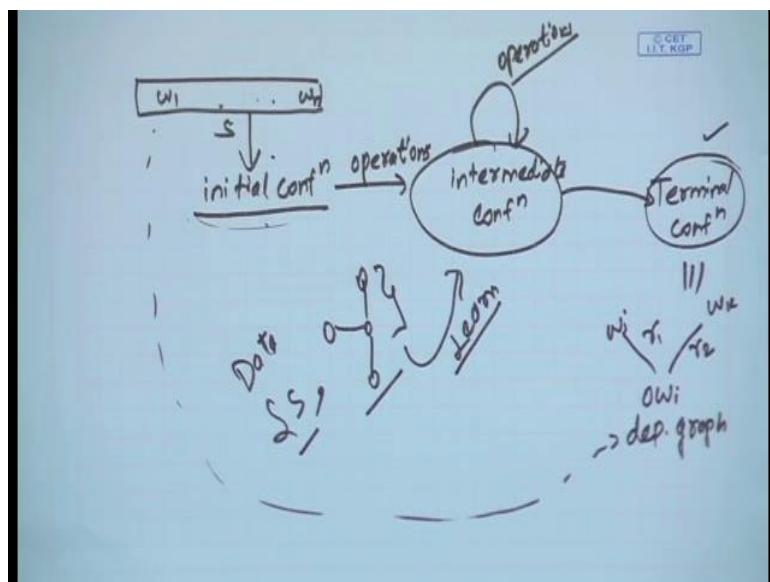
Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 29**  
**Transition Based Parsing: Learning**

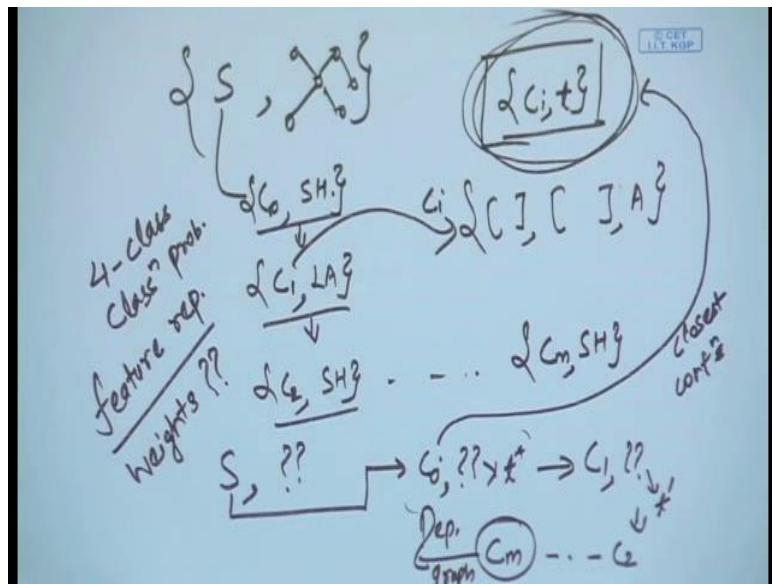
Welcome back for the third lecture of this week. So, in the last lecture, we had defined what is the notion of our transition based parsing. And we saw what are the configurations that I should have what are transitions I can take and how do I come up with a final dependency graph. And we took an example and showed what are the transitions you can take and how you should be taking the transitions. And then we ended with saying that how I will be using that for getting parse for a new sentence. So, this is something that we had initially asked.

(Refer Slide Time: 00:58)



So, I have some sentence  $S$ ,  $S$  I can find out what initial configuration is why the function. I keep on taking some transitions go to some intermediate configuration until I obtain terminal configuration. And I said with the data I will have some sentences and their corresponding dependency parsing. And from there I will try to learn what are the operations of transitions I am where to take. Now, in this lecture we will see for a particular problem how we will be doing the learning part.

(Refer Slide Time: 01:48)



So, again let me start by giving you the basic intuition. So, what will be the basic intuition? So, in the last lecture, you had learned that for a sentence if you know the dependency graph so this is my labeled data, if you know that, you can run through the steps very, very easily, the steps of transitions. So, you can say that this is my initial configuration. And you should find out what is the transition I should take, say it is shift. So, you can store somewhere, this my initial configuration  $C_0$  and this is the transition. Then by taking this transition, you will go to some other configuration  $C_1$ , what is the transition it may be left-arc again some  $C_2$ , it can be shift and so on, some  $C_m$ , it can be shift or something.

So, this given a sentence, you can easily run through these steps and find out. And what is this  $C_i$ ?  $C_i$  is nothing but a some words in stack, some words in buffer and something in my arc that is my  $C_i$ . So, that means, suppose I am given a set of sentences and their dependency graph, I can store all the possible set of configuration and all the possible by all the possible I mean whatever I obtaining from these sentences and their corresponding transitions. And this I can have a last set. So, I will have a set of all possible  $C_i$  and that optimal transition that I should be taking and  $C_i$ 's of this form something in stack and something in buffer something in arc.

Now, what is my problem at run time? At run time I am given a sentence  $S$ , I do not know its dependency parse. So, I start my transition, I converted to some initial configuration  $C_0$  that is easy we have the function of converting to initial configuration. There I have to find out

what is the transition I should take. Now, what will be the idea? I will try to use this set of data that I have. I know for what configurations, what transitions word taken in my gold standard or in my set of labeled data set. So, I will try to find out. So, before going into what this is the machine learning approach, we will use, so what will be intuitive idea try to find out what are the closest configurations in the set; And for those closest configurations, what transition was taken and I will try to use that t star from there.

Suppose I find out the t star is the transition that was taken to the closest configuration here. So, I will use t star. And once I know this t star, I can transit it to next one C 1; again the question will come what is the transition I should take. So, again go back to this and choose t dash, take this go to C 2 keep on doing that until you will come up with the final configuration C m, and that is why you say this is the dependency graph for S. And this is the intuitive idea. In the last lecture, we have seen how do come up with this set. And today we will see, how do I approach this problem, so that I can come up with this function that what is the closest configuration from here to here, what is the transition that I should be taking.

So, one important idea that we had already discussed earlier in this course is that for example, how do you find out the closest configuration, and what is the transition that we are taking. This you have to use by using some sort of classifier that is you are trying to classify for a given configuration among all the four transitions which transitions will be taken. So, you are treating it as a four class classification problem; at each point you are trying to classify among one of the four classes. And how are you going to classify? You have to convert your input data in some representation. So, this will be using some feature representation. So, you will have to convert your configuration into some sort features, feature vector. And for those features, you have to learn the weights. And this weight you have to learn by the training examples again. And once you have learn the optimal weights, you can find out what is the transition at any given point and this is what we will be discussing in this lecture.

(Refer Slide Time: 07:12)

The slide has a blue header bar with the title "Classifier-Based Parsing". Below it is a white content area divided into three sections:

- Data-driven deterministic parsing:**
  - Deterministic parsing requires an **oracle**.
  - An oracle can be approximated by a **classifier**.
  - A classifier can be trained using **treebank** data.
- Learning Problem**

Approximate a function from **configurations**, represented by feature vectors to **transitions**, given a training set of gold standard **transition sequences**.
- Three issues**
  - How to represent configurations by feature vectors?
  - How to derive training data from treebanks?
  - How to learn classifiers?

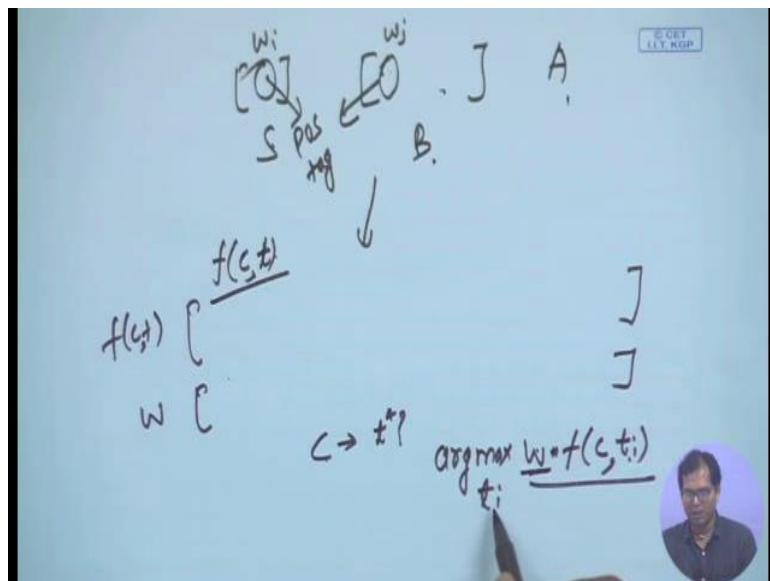
At the bottom of the slide, there is a footer bar with the following information:  
Ptwan Goyal (IIT Kharagpur)   Transition Based Parsing: Learning   Week 6, Lecture 3   2 / 8

So, let us see. So, now we are talking about the data driven deterministic parsing. So, I have written here certain things like deterministic parsing requires in oracle. What do I mean by oracle? So, oracle is nothing but the set of configurations, and the set of transition that I took. So, this you have already seen in the last lecture, how do you find out configuration and transitions. Now, what we are going to do? We want to approximate it by a classifier. So, we will be learning classifier from there, and it will be trained using the treebank data. So, whatever data we have in the gold standard label data, we will use that to train our classifier also. So, you will use that label data for two different tasks; first for building the oracle configuration, transition, configuration, transition; second to learn the weights of my classifier.

Now, what is the learning problem? Now, as we said we will be given a configuration as an input and we want to find out what is the transition I should be taking at a particular configuration. So, ideally I want to approximate the function that takes a configuration which is represented by a feature vectors, two transitions - so configuration to transitions. Here configuration is nothing but a feature vector form, because otherwise how do you compare between two configurations, so that is why you will take give it a very abstract representation in terms of some feature vector form. And you will learn a function from feature vector to the optimal transition, and this will be your classifier. And how will you learn that you will be given a training set of gold standard transition sequences that we already have seen.

So, now to completely solve this problem or to completely understand this problem there are three issues that you need to understand. First is how do I represent configuration by a feature vector. Second, how do I derive training data from my treebanks, and third is how do I learn my classifiers. So, let us try to answer each of these three questions. So, how do I represent configuration by feature vectors, and this is something that we had done earlier in the class that how do I convert a given state or represent to a feature vector.

(Refer Slide Time: 09:58)



So, what is my configuration? My configuration is nothing but a stack, buffer and arcs. And I want to convert that to some feature vector. And feature vector again here we will take it a very simple form like  $f(c, t)$ , it is a function over the configuration and the transition. And we will try to define features independent of transition, so each feature that we define can have four copies four, four different transitions, so  $f(c, t_1), f(c, t_2), f(c, t_3), f(c, t_4)$  like that. So, what are the different things that I can use in feature? So, I can use things like what is the word at top of this stack, what is the word at top of buffer, they are very important, what is the word here.

Then I may want to use what is the part of speech tag of these words, because sometimes some relations would might directly dependent on whether it is a verb and it is a noun then there might be a relation otherwise not. So, I might use the part of speech tag. I might go to the lemma; I might want to use what is the distance between this word and this word in the actual sentence. I might also want to use what are neighbors here; I might want to use what

are relations that I have already been established with this word or this word, so that means I will define certain conditions over stack, buffer and arcs and that will I would like to take my features. And again these can be some binary questions that I am asking. So, that is the distance between these two words between 2 to 5, yes or no. So, like that these can be my condition that is my features, and I will define it for all the four transitions.

(Refer Slide Time: 12:01)

A feature representation  $f(c)$  of a configuration  $c$  is a vector of simple features  $f_i(c)$ .

*Typical Features*

- Nodes:
  - Target nodes (top of  $S$ , head of  $B$ )
  - Linear context (neighbors in  $S$  and  $B$ )
  - Structural context (parents, children, siblings in  $G$ )
- Attributes:
  - Word form (and lemma)
  - Part-of-speech (and morpho-syntactic features)
  - Dependency type (if labeled)
  - Distance (between target tokens)

So, let us look at what are the different feature models we can take in general. So, I am representing configuration  $c$  by a vector of simple features. So, like I can use the nodes. So, what is the top of this stack, what is the head of buffer, I can use the linear context that is what are the neighbors in  $S$  stack of the top word, what are the neighbors in buffer of the top word. Then I might also use what are the parents, children, siblings depending on what relations are already been established in my set of arcs.

Then I can go to some other attributes like I can use the word form, I can use also its lemma. And we can use them up part of speech tag and various other features for example, is the word on top of stack ends with ed or ing things like that. And I might be able to use the dependency type if I am handling a labeled dependency problem. So, just a word what do I mean by labeled dependency problem that means, I also want to find out for two words what is the dependency relation label. And if I am solving unlabeled problem that means, I want to just want to establish a relation, but I am not concern with the actual dependency type.

So, I am not worried about putting the label on the arc, but just the structure of the tree. So, if I am solving label problem, I might also have to see what is the relation type that I am establishing. And we will also see the distance between different tokens as one of the features. So, these are some typical examples, but it does not mean that you are limited only to using this set of features. And as I keep on seeing for your particular task, you might have to think what might be some interesting features that you can use. So, by using all these features, I am putting my binary questions, I can represent my configuration as in terms of a feature representation. So, this is my first question. Now, what was the second question?

(Refer Slide Time: 14:08)

## Deterministic Parsing

To guide the parser, a linear classifier can be used:

$$t^* = \arg \max_t w \cdot f(c, t)$$

Weight vector  $w$  learned from treebank data.

*Using classifier at run-time*

```

PARSE( $w_1, \dots, w_n$ )
1    $c \leftarrow ([], [w_1, \dots, w_n]_B, [])$ 
2   while  $B_c \neq []$ 
3        $t^* \leftarrow \arg \max_t w \cdot f(c, t)$ 
4        $c \leftarrow t^*(c)$ 
5   return  $T = (\{w_1, \dots, w_n\}, A_c)$ 

```

Pawan Goyal (IIT Kharagpur) Transition Based Parsing: Learning Week 6, Lecture 3 4 / 8

$$\begin{aligned} t^* \\ = \arg \max_w w \cdot f(c, t) \end{aligned}$$

Now, how do I use this at run time, but actually find the parse of the sentence; idea would be something like that. So, let me start from here. At run time, you are given a sentence  $w_1$  to  $w_n$ . And you can always go to the initial configuration, where the  $s$  stack is empty, buffer contains all the words and arc is empty. Now, this is your configuration and you are in a loop while the buffer is not empty, you keep on taking some transitions. Now, this is the task say run time, so this configuration  $c$  you know how to convert to the to a feature vector because you are defined your features. So, you can ask the questions at this point and find out your vector  $f(c, t)$ .

Now, what is my classifier? My classifier is simple. I have learned the weights of my features assume that I have learned, we will see how to learn the weights. So, once I have learned the weights of my features, I will multiply the weights with  $f(c, t)$  and find out what is the particular transition that is giving the maximum value that means, I know what is my  $f(c, t)$  feature vector, and I know this is my  $f(c, t)$  and this is my weights. So, now at run time, I am given a configuration  $c$  and I need to find out what is the optimal transition how do I do that. An idea is that I will multiply  $w$  with  $f(c, t)$  for all the four transitions. So,  $t_i$  is shift, left-arc, reduce and right-arc. And I will take which one gives the maximum value  $\text{argmax}$  over  $t_i$ .

So, at run time, any configuration, I can convert to  $f(c, t)$  very easily. I will already have the weight vectors the only thing that I have to do multiply the weight vector with the feature vector find out four different values four different transitions choose the maximum or choose the transitions that has gives the maximum value that is the transition you will take. And then if you go back this is the transition you take and then apply this transition over this configuration to find the next configuration, and keep on going in this loop until your buffer is empty, and this is what your run time. Now, what is not clear to you right now is how do we learn these weights, everything else is clear.

(Refer Slide Time: 17:12)

*Training data*

- Training instances have the form  $(f(c), t)$ , where
  - $f(c)$  is a feature representation of a configuration  $c$ ,
  - $t$  is the correct transition out of  $c$  (i.e.,  $o(c) = t$ ).
- Given a dependency treebank, we can sample the oracle function  $o$  as follows:
  - For each sentence  $x$  with gold standard dependency graph  $G_x$ , construct a transition sequence  $C_{0,m} = (c_0, c_1, \dots, c_m)$  such that
 
$$c_0 = c_s(x),$$

$$G_{c_m} = G_x$$
  - For each configuration  $c_i$  ( $i < m$ ), we construct a training instance  $(f(c_i), t_i)$ , where  $t_i(c_i) = c_{i+1}$ .

Pawan Goyal (IIT Kharagpur)      Transition Based Parsing: Learning      Week 6, Lecture 3      5 / 8

So, let us see. So, learning weights we will have to use the labeled data that we have is the training data. Now, let us see what are the steps that we need to do over the training data. Now, training data, I will have the instances of this form  $f(c)$  and  $t$  is this clear entering data

we will have configurations and transitions, yes. And configuration I know what is the feature representation, so I convert the feature vector. So, I can have  $f c$  and  $t f c$  is nothing, but the feature representation of the configuration  $c$  and  $t$  is the correct transition out of it starting from  $c$ . And this I can obtain from my oracle. Remember in oracle I had my configuration and the optimal transition, so I know this configuration what is the transition it should take. So, from there, I go to next step  $f c$  and  $t$ .

Now, this is something that we have done in the last class, but let me try to repeat that again that how do we sample this oracle function from the set of labeled sentences labeled of dependency graph. So, for each sentence  $x$  with the gold standard dependency graph  $g_x$ , you have to construct a transition sequence right like we did in the last class for the example he sent her a letter, such that  $c_0$  is something that will be obtained by applying the initialization function on  $x$  and this is the final dependency graph.

Now, for each intermediate configuration we will construct a training instance. So, right we will have  $c_i t_i c_i t_i$  and  $c_i$  will go to  $f c_i$  and this is the condition for how am I moving in from one configuration to another configuration. So, this is the same thing that I have discussed earlier in today's lecture that what is the idea is starting from the gold standard sentence and dependency graph find out this sequence  $c_i t_i c_i t_i$ . Now, the only addition here is I convert each  $c_i$  to its feature representation, so that is why I have  $f c_i t_i$ .

(Refer Slide Time: 19:17)

*Standard Oracle for Arc-Eager Parsing*

$o(c, T) =$

- **Left-Arc** if  $\text{top}(S_c) \leftarrow \text{first}(B_c)$  in  $T$
- **Right-Arc** if  $\text{top}(S_c) \rightarrow \text{first}(B_c)$  in  $T$
- **Reduce** if  $\exists w < \text{top}(S_c) : w \leftrightarrow \text{first}(B_c)$  in  $T$
- **Shift** otherwise

Pawan Goyal (IIT Kharagpur) Transition Based Parsing: Learning Week 6, Lecture 3 6/8

And how do you sample the transitions in oracle this is something again that we did in the last class. So, if you see that in my dependency graph the current top of the word in the stack is connected to the top of the word in buffer. So, you will have a relation depending on the direction it will be left-arc or right-arc. So, here if the top of the word in buffer is the head and this is the dependent you make a left-arc transition. So, this is what you will store.

If top word in the in the stack is head then you will have a right-arc relation, yes. Then how do you choose between reduce and shift, if there is a word below that of the stack such that it is connect to the first word in the buffer, then you do reduce otherwise you shift. And remember this is rule of thumb that we discussed in the last lecture. If you are choose between reduce and shift this is the condition that you can use. So, I hope the idea is clear, you are starting with the sentence in training data, you have the gold standard dependency graph, you keep on going through your transitions and store it somewhere f c i t i f c i t i f c i t i.

(Refer Slide Time: 20:40)

*Online Learning with an Oracle*

```

LEARN({ $T_1, \dots, T_N$ })
1    $w \leftarrow 0.0$ 
2   for  $i$  in  $1..K$ 
3     for  $j$  in  $1..N$ 
4        $c \leftarrow ([], [w_1, \dots, w_{n_j}]_B, [])$ 
5       while  $B_c \neq []$ 
6          $t^* \leftarrow \arg \max_t w.f(c, t)$ 
7          $t_o \leftarrow o(c, T_i)$ 
8         if  $t^* \neq t_o$ 
9            $w \leftarrow w + f(c, t_o) - f(c, t^*)$ 
10           $c \leftarrow t_o(c)$ 
11   return  $w$ 

```

Oracle  $o(c, T_i)$  returns the optimal transition of  $c$  and  $T_i$



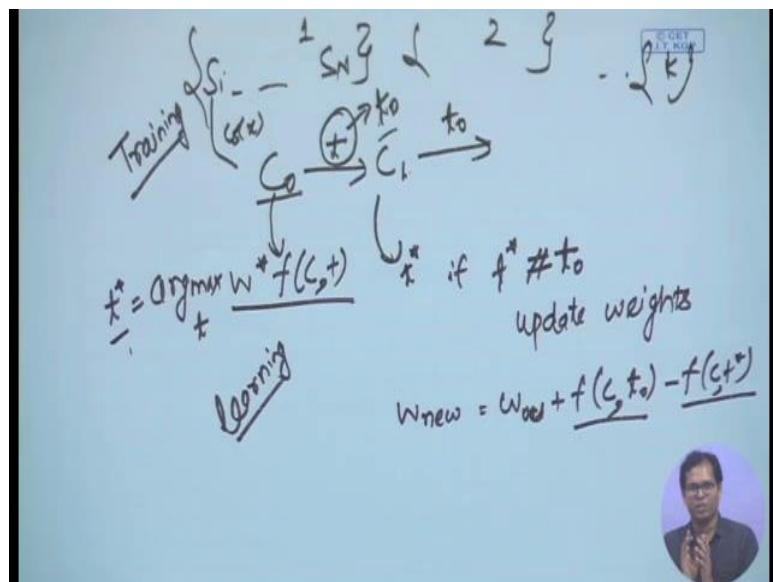
Pawan Goyal (IIT Kharagpur)      Transition Based Parsing: Learning      Week 6, Lecture 1

Now, how do I use that to learn the weights? Now, and this is the idea of learning the weights. So, what you will do? You will start with some initial set of weights. So, here I have said all the weights can be initialized to 0, but probably not will 0, you can initialize with some other numbers say some initial random numbers or some uniform numbers, you initialize your weights. Now, what are you going to do? So, there are two loops here for  $i$  is 1 to  $K$  this is the number of iterations that you are doing; for  $j$  in 1 to  $N$ , 1 to  $N$  is the all the set

of sentences that you have in your training data. So, you are doing multiple iterations over your training data.

In each iteration, what do you do, you take the sentence, yes, you get the initial configuration fine. Now while buffer is not empty, so what you are doing right now, you are again repeating the same stuff over each sentence in the training data. You start with the initial configuration and now try to find the transition as per your current weights, so that is where the idea of learning comes in.

(Refer Slide Time: 22:05)



So, let me try to explain it here. So, you have a sentence  $S$ , now you can apply the  $c_s x$  function or and you go to initial configuration  $C_0$ . Now, you take a transition to go to  $C$ . Now, this is your sentence in your training data that means, you know what is the transition you should take, yes, but I want to use this idea to learn and how do I do that. Now, suppose that you are actually at the run time at testing time, so then you do not know the transition. So, you will convert that to some feature vector  $f(c, t)$ . Now, at run time, how do you find out the optimal transition multiplied with the weights and take the argmax. And let us say this is  $t^*$ , this you can do that even if it is in the training set and let us call it  $t^{naught}$  - optimal transition.

Now, what is the idea? You are still in the learning stage. So, your weights will not be optimal. So, when you do this operation, you may not get the optimal transition, you may get something else and that is where you will try to adopt your weights. So, you will say if  $t^*$

not equal to  $t_0$ , then you update your weights. And how will you update your weights such that you go in the direction of the actual transition, and away from the transition of the transition that you obtain at the current point. So, simple thing is  $w_{\text{new}}$  would be  $w_{\text{old}}$  plus  $f_{c,t} - f_{c,t^*}$ . So, going in the direction of the optimal transition and away from the transition that you are currently predicting, and there can be some learning rate and all that we are not discussing right now. So, there will be some learning weights by which you will do this update. So, you will have new weights.

Again you keep on doing it for  $c_1, c_2$  you know what is the transition optimal transition, but you will find out  $\text{argmax } t^*$  match with this; if they are not the same, you will again update your weights. So, you will keep on doing that for all the sentences  $S_1$  to  $S_N$  in your training set and you will do it 1, 2 some  $K$  times until the weights are converging. So, once the weights converged, we stop. So, this is what we have shown here. So, you start for each sentence you have some initial configuration, while buffer is not empty. So, you keep on doing the stuff. Find out what is the optimal transition as per your weights, find the optimal transition from the oracle; if they are not matching, update your weights, but you take the correct transition. So, the next time you are starting with correct configuration, keep on repeating it. And finally, you will end up with new set of weights.

(Refer Slide Time: 25:50)

*Example*

Consider the sentence, 'John saw Mary'.

- Draw a dependency graph for this sentence.
- Assume that you are learning a classifier for the data-driven deterministic parsing and the above sentence is a gold-standard parse in your training data. You are also given that *John* and *Mary* are 'Nouns', while the POS tag of *saw* is 'Verb'. Assume that your features correspond to the following conditions:
  - The stack is empty
  - Top of stack is Noun and Top of buffer is Verb
  - Top of stack is Verb and Top of buffer is Noun

Initialize the weights of all your features to 5.0, except that in all of the above cases, you give a weight of 5.5 to *Left-Arc*. Define your feature vector and the initial weight vector.

- Use this gold standard parse during online learning and report the weights after completing one full iteration of Arc-Eager parsing over this sentence.

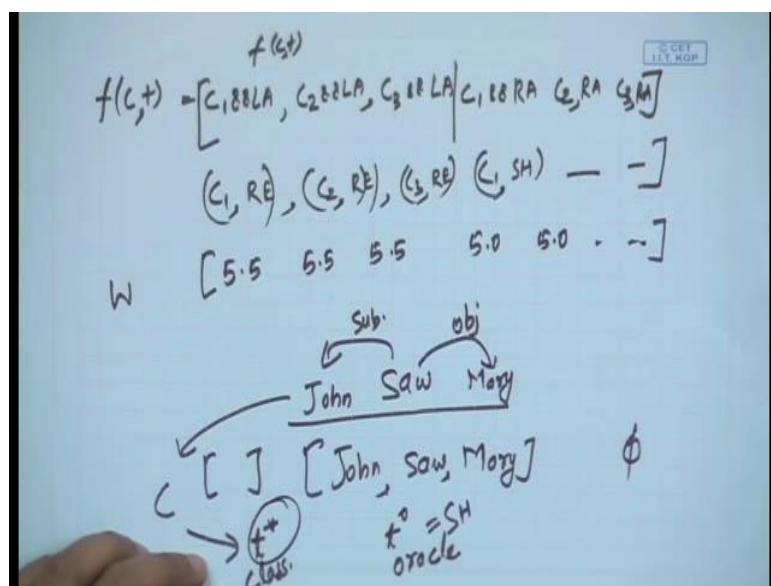
Pawan Goyal (IIT Kharagpur)      Transition Based Parsing: Learning      Week 6, Lecture 3      8 / 8

So, now to further understand that let us take an example and that will make it clear to you that how the learning or how the weight updation takes place. So, this is simple example. So,

I have a sentence John saw Mary. So, what do you need to do? First question is draw a dependency graph for this sentence that is very easy, yes. Now, the next question says that you are using the data driven dependency parsing the same method that we have discussed in this lecture and in the last lecture. And you already have the gold standard parse in your training data and you have some other information like John and Mary are nouns; and saw is verb, and also given you features. And you are told that initialize your weights to 5 except that for left-arc the weights are 5.5 define your feature vector, and the initial weight vector.

So, let us try to do this. So, how many conditions are we seen, we are seeing three conditions over my configuration. The stack is empty, top of stack is noun and top of buffer is verb, top of stack is verb and top of buffer is noun, three conditions. Now, these three conditions I have to check for all the four different transitions. So, what is the size of my feature vector 3 into 4 - 12?

(Refer Slide Time: 27:15)



So, my feature vector, so it is of 12 dimension. And what are my features, so first feature let me write it simply condition one that is the stack is empty and same as starting with left-arc; transition is left-arc. Second feature can be condition 2 and left-arc; third condition three and left-arc, yes. This is my  $f_c, t_a$  condition over the configuration and transition. First three elements next three elements same  $c_1$ , but now transition will change still right-arc;  $c_2$  right-arc,  $c_3$  right-arc. And then the next elements  $c_1$  reduce,  $c_2$  reduce,  $c_3$  reduce, and here  $c_1$  shift,  $c_2$  shift,  $c_3$  shift. So, this is my feature vector twelve elements here. Now,

what is my weight vector? Initial weight vector we said all the elements are 5 except the left arc is 5.5. So, the weights are 5.5, 5.5, 5.5 and everything else is 5.0, 5.0 and so on that is your initial weight vector and your task is now so this was the first question what is my dependency parse John saw Mary. So, saw is here, John and Mary this is subject and this is object.

Now let us see what the question says further. So, the next question says use this gold standard parse during online learning and report the weights after completing one full iteration of arc eager parsing. So, it says that now you have to learn the weights using the arc eager parsing or the transition parsing that we have seen.

So, now let us see how do we learn the weights. So, this is what we have defined right now. We have this features, we have this weights initially as per dependency graphs. So, how will I start learning? In learning, I will take this sentence I will put it to the initial configuration initial configuration is what this stack is empty, buffer contains John, saw and Mary; and arc is empty. So, now, at this configuration C, I have to choose what is the optimal transition as per my classifier. And I have to choose the optimal transition as per my oracle from oracle what will be t 0. Oracle I will be very easily saying that this would be shift I am doing a shift at this point I should shift here, but what is being predicted by my classifier. So, let us see what will be t star as per my classifier.

(Refer Slide Time: 30:55)

Handwritten notes for arc-eager parsing:

$$t^* = \underset{t}{\operatorname{argmax}} \omega \cdot f(c, t)$$

$$f(c, LA) = [1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0] \quad \text{X}$$

$$f(c, RA) = [0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0] \quad \omega \cdot f(c, LA) = 5.5$$

$$SH = [ \quad \quad \quad 1 \ 0 \ 0 ] \quad \omega \cdot f(c, RA) = 5.0$$

$$RE \quad \quad \quad \omega + f(c, t^*) - f(c, t^*) \quad " \quad SH = 5.0$$

$$" \quad \quad \quad \quad \quad \quad " \quad RE = 5.0$$

$$t^* = LA \quad t^0 = SH$$

$$\omega = [4.5 \ 5.5 \ 5.5 \ | \ 5.0 \ 5.0 \ 5.0 \ | \ 5.0 \ 5.0 \ 5.0 \ | \ 6.0 \ 6.0 \ 6.0]$$

So, for my classifier how do I obtain  $t^*$ ? So,  $t^*$  would be  $\text{argmax}_t w \cdot f(c, t)$  for all possible transitions. So, let us do one-by-one. So, what will be  $f(c, LA)$ ? What is the feature vector when the transition is LA? So, for that let us look at my feature vector definition. So, this will be a binary value at each point c 1 and LA. What is c 1 top of the stack is this stack is empty, so c 1 is 1, and transition is left-arc, so this will be 1. C 2 top of stack is noun, and top buffer is verb. Now, top of stack is empty it cannot contain a noun, so c 2 is 0; so already this will be 0. C 3 again say top of stack is verb and top of buffer is noun again this will be 0 that is why I fill in my feature vector

Now, let us go to this. Now, immediately as you move to some other transition RA this should be 0, yes because here your transitions is LA. So, everything else will be 0, 12 elements. Now, what is your weight vector? Weight vector is here, say if you multiply weight vector with  $f(c, LA)$ , what do you get? So,  $w \cdot f(c, LA)$  is equal to 1. So, only one element is 1. So, I will multiply with that this will give you 5.5. Now, similarly now you can easily figure out what are the other features  $f(c, RA)$ . For RA, similarly only this element will be 1, everything else will be 0. So, what is  $w \cdot f(c, RA)$  that will be 5. And similarly, if you keep on doing for all shift and reduce you will find this for shift is 5 and this for reduce is 5, yes.

So, now what is your  $t^*$ ?  $\text{argmax}_t w \cdot f(c, t)$  that is will be left-arc. And what is your  $t_0$  optimal is shift as per your oracle. And how do you learn your weights, if  $t^*$  is not the same as  $t_0$ , you will update your weights. And what is the (Refer Time: 33:45) update  $w$  plus  $f(c, t_0) - f(c, t^*)$ . So, what is  $f(c, t_0)$ ? That is  $f(c, SH)$ . So, what will be this function? So, suppose SH was at the end. So, it was 1 0 0 and everything else is 0 and this is my LA. So, what will be the new weight vector? So, I have the original weight vector that is this 1 plus this minus this. So, what will be the new weight vector?  $t$  will be I am subtracting one here, so 4.5, 5.5, 5.5, 5.0, 5.0, 5.0, 5.0, 5.0. And now I come to shift in shift I am adding that 1, so it will be 6.0, 5.0, 5.0, and that is your new weight vector.

And now you will work with this weight vector for the next set of configuration. So, what will be the next configuration from here you will apply shift and next configuration will be John saw Mary, and phi. Again you will convert it to the feature vector see what is  $t^*$  that you are getting what is  $t_0$  if they are not matching update your weights and that you will continue until you arrive at the terminal configuration. And then you will have the final weight vectors. So, I will encourage all of you that you should try it this full example on your

own, and see what is the final weight vector that you are getting. And even if you are trying to see that by using this weight vector, does that help in that now with the new weight vector if you try it on the old configuration you will be closer to the optimal configuration as per the oracle and not what your (Refer Time: 36:10) early updating.

So, this is the idea of how you can use the machine learning methods for this dependency parsing by taking this example of arc eager of transition based parsing. Now, in the next lecture, we will start discussion on a new method of dependency parsing, and we will see that again how we can use the label data for doing this.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 30**  
**MST - Based Dependency Parsing**

Welcome back for the fourth lecture of this week. So in the last 2 lectures we had discussed a particular (Refer Time: 00:26) dependency parsing that was a transition based parsing method. And we had discussed how we can formulate parsing as a particular problem where we start with an initial configuration and moving from one configuration rather by using various transitions. And we modeled or learned these transitions using some training data that we already had. So this was a data driven approach.

Now, in this lecture we will talk about another data driven approach here. So the one main difference from the previous one is that we do not assume that projectivity constraint over the dependency graph. So in the last method that we had covered we were assuming that the dependency graph that we want should be projective although later on there are some variations proposed where this constraint is not required. So you might if you want to look into those.

So coming to what we are going to discuss in today's lecture and in the next lecture. So this is the approach using where we are formulating dependency parsing as a problem of finding maximum spanning tree.

Now, what is the basic idea? So we are starting with the sentence that is given to me. So this is the sentence for which I want to find the dependency graph. And how do we formulate the problem. Firstly, think about the sentence as a set of nodes these are the words and assume that all the possible nodes are connected to each other. We will also include additional node called root so that we can find out what is the main head of this sentence now. Once we start by assuming all possible connections my problem is to find out what is the maximum spanning tree from this all the connections that I can build. And this I would assume corresponds to my dependency graph. And the weights etcetera that will define for this graph will be learned by using some training data that will be available to me.

(Refer Slide Time: 02:35)

### Maximum Spanning Tree Based

*Basic Idea*  
Starting from all possible connections, find the maximum spanning tree.

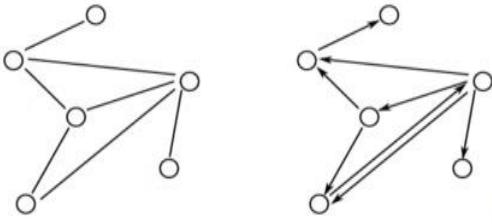
Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing Week 6, Lecture 4 2 / 16

So the main idea here is I am given a dependency graph. So I start with a sentence John saw Mary. And I assume an additional node like root. Assume that initially everything is connected. So from root there are all possible connections to all that node. And each node has an incoming and an outgoing edge with respect to every other node. How do we learn these edge weights? This is something that we will discuss, but assume that the weights are already given to you. Now your problem would be how do I find out what is the maximum spanning tree from this graph. And this suppose this is the maximum spanning tree then this is what I will assume corresponds to my dependency graph. So my problem is starting from here to come up with the maximum spanning tree that might be dependency graph.

(Refer Slide Time: 03:24)

### Some Graph Theory Reminders

- A graph  $G = (V, A)$  is a set of vertices  $V$  and arcs  $(i, j) \in A$  where  $i, j \in V$ .
- Undirected graphs:  $(i, j) \in A \Leftrightarrow (j, i) \in A$
- Directed graphs (digraphs) :  $(i, j) \in A \Rightarrow (j, i) \notin A$



Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing Week 6, Lec. 16

So now starting with some simple reminders from graph theory on what is a maximum spanning tree. So you are now familiar with what is a graph. So when I say graph this is a set of  $V$  that is set of vertices and  $A$  set of arcs. So arcs generally connect to different nodes in the graph. So arc is connecting to nodes  $i$  and  $j$  both  $i$  and  $j$  are in the set  $V$ . Now if I talk about undirected graph like the figure on the left. So if I say that  $i$  and  $j$  are in the set of arcs then  $j$  and  $i$  are also there is no directions in this in this set of arcs.

But if I see that my graph is directed. Then if  $i$  and  $j$  are in the set of arcs then  $j$  and  $i$  may not be in the set of arcs. Like here I have a connection from this node to this node, but no connection from this node to this node.

(Refer Slide Time: 04:26)

### Multi-Digraphs

- A multi-digraph is a digraph where multiple arcs between vertices are possible
- $(i, j, k) \in A$  represents the  $k^{\text{th}}$  arc from vertex  $i$  to vertex  $j$ .

Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing Week 6, Lecture 4 16

So what is a simple graph and what is a directed graph? Directed graph can also be written as a digraph. Now we can also see what is a multi-digraph. A multi digraph what would happen between a set of vertices? There might be more than one possible arc. Like here you are seeing between this node and this node there are 3 different arcs. So that is why you will have an additional index saying  $i$ ,  $j$  and  $k$  between the nodes  $i$  and  $j$  what is the  $k^{\text{th}}$  arc. So it is similar to the previous one except that between 2 vertices you can have more than one arc.

(Refer Slide Time: 05:12)

### Directed Spanning Trees

- A directed spanning tree of a (multi-)digraph  $G = (V, A)$  is a subgraph  $G' = (V', A')$  such that:
  - $V' = V$
  - $A' \subseteq A$ , and  $|A'| = |V'| - 1$
  - $G'$  is a tree (acyclic)
- A spanning tree of the following (multi-)digraphs

Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing Week 6, Lecture 4 16

Now, once we know what is a multi-digraph, let me define the notion of what is the directed spanning tree of the multi digraph. So once I start with the multi digraph that is set of nodes  $V$  and there is a sub graph  $G'$  that is some  $V'$  prime a prime is called a directed spanning tree if the following conditions are followed. What are the conditions? In the directed spanning tree, the number of the set of nodes are the same as the set of nodes in the original graph.

So I will have to take all the nodes from the original graph. So now, what do I do with the edges? So the set of arcs that I take in  $G'$  is the subset of the original set of arcs with the constraint that the number of edges that I have now is equivalent to number of nodes minus 1 and the third condition is the graph  $G'$  is a tree. It is there is no cycle in the graph. So let us take example.

So I have 2 different multi digraphs that are given in the first and second figure. And you are being shown a directed spanning tree for this multi digraphs. So what do you see here? So it has the same number of nodes as were there in the original multi digraph and what about the number of connections. So you see there are 1 2 3 4 and 5 connections and number of nodes are 1 2 3 4 5 6. So number of connections are exactly number of nodes minus 1. And you can take any possible any possible 5 connections in among these nodes such that there is no cycle and we called a directed spanning tree.

(Refer Slide Time: 07:14)

### Weighted Directed Spanning Trees

- Assume we have a weight function for each arc in a multi-digraph  $G = (V, A)$ .
- Define  $w_{ij}^k \geq 0$  to be the weight of  $(i, j, k) \in A$  for a multi-digraph
- Define the weight of directed spanning tree  $G'$  of graph  $G$  as

$$w(G') = \sum_{(i,j,k) \in G'} w_{ij}^k$$


Pawan Goyal (IIT Kharagpur)      MST-based Dependency Parsing      Week 6, Lecture 6

Now, once we know what is directed spanning tree we can also define what is a weighted directed spanning tree. So here what we will do? In my graph with all the arcs I will also associate a weight. So each arc is now labeled or weighted by some numerical value and this I will call at  $w_{ijk}$  for the  $i$  between the vertices  $i$  and  $j$  what is the weight for the  $k$ th arc. So in  $w_{ij}^k$  superscript  $k$  is the weight of the edge  $i j k$ . Now I will also define what is the weight of my directed spanning tree. So this will be simply summation over the weights of all the edges in my directed spanning tree. So from my multi digraph i find a directed spanning tree definition was already covered in the previous slide.

Now, whatever edges I am obtaining I will find out the weights for each of these and sum over them. And that will define the weight of my directed spanning tree. Now the idea here is for a given starting from a given multi digraph you can obtain a number of different directed spanning trees. So your problem is to find out which of these has the maximum weight. So now, I by this concept of weighted spanning tree I have also define what is the weight of the directed spanning tree. So now, among all the possibilities I will find out the one that is having the maximum weight.

(Refer Slide Time: 08:47)

**Maximum Spanning Trees (MST)**

Let  $T(G)$  be the set of all spanning trees for graph  $G$

The MST problem

Find the spanning tree  $G'$  of the graph  $G$  that has the highest weight

$$G' = \arg \max_{G' \in T(G)} w(G') = \arg \max_{G' \in T(G)} \sum_{(i,j,k) \in G'} w_{ij}^k$$

Pawan Goyal (IIT Kharagpur)      MST-based Dependency Parsing      Week 6, Lecture 16

So it is starting from the graph  $G$  let us say  $T(G)$  is the set of all the possible spanning trees that I can obtained. So like this is this is an example this multi digraph is given and there are these 4 possible spanning trees. So these are all directed spanning trees. Now I have to find out which of these has the maximum weight. So this is my MST problem.

Finding the spanning tree  $G'$  of the graph  $G$  that has the maximum weight and the weight is written simply as summation over the weights of all the edges of that graph. So I find out all the edges of that graph, sum over the weights and that will give me the weight of the final graph.

So now my problem is from my sentence, whenever I construct a multi digraph that is all the possible connections, a fix set of directed spanning trees are possible. Now among those which is having the highest weight and that is the one that I will say as my dependency graph. So now, there are many questions like how do I convert a sentence to an initial configuration. Analogous to what we did in the previous method. Then the important problem here is how do we define the weights of my edges. And once I define the weights how do I choose the maximum spanning tree. So these are 3 different problems and then that is what we will be studying in this lecture and in the next.

(Refer Slide Time: 10:25)

### Finding MST

*Directed Graph*

For each sentence  $x$ , define the directed graph  $G_x = (V_x, E_x)$  given by

$$V_x = \{x_0 = \text{root}, x_1, \dots, x_n\}$$

$$E_x = \{(i, j) : i \neq j, (i, j) \in [0 : n] \times [1 : n]\}$$

*$G_x$  is a graph with*

- the sentence words and the dummy root symbol as vertices and
- a directed edge between every pair of distinct words and
- a directed edge from the root symbol to every word

Pawan Goyal (IIT Kharagpur)      MST-based Dependency Parsing      Week 6, Lecture 4      8 / 16

So now for the sentence how do I find the maximum spanning tree? So let us see this is the first problem how do I convert the sentence to the initial configuration from where I can start, so what is that? For each sentence  $x$  you define the directed graph  $G_x$  as  $V_x E_x$  that is given by  $V_x$  contains and additional load like root and all the all the words that occur in my sentence  $x$ . And what are my edges I have all possible edges except for the same node I do not have any self-edge from the node to itself.

But I have edges from every node to every other node except that the word the root node will not have any incoming edges. That is why it is written here the edges are from the node 0 to n to one to n 0 denotes the root node from root there are only outgoing edges and for every other node they have both incoming edges and outgoing edges. So that is my  $G_x$  is a graph, where the sentence words and the dummy root symbol are the vertices. This is my set of vertices and there is a directed edge between every pair of distinct words yes that is what we have seen here. And a directed edge from the root symbol to every word from 0 to 1 to n this is my initial graph.

(Refer Slide Time: 11:54)

*Chu-Liu-Edmonds Algorithm*

*Chu-Liu-Edmonds Algorithm*

- Each vertex in the graph greedily selects the incoming edge with the highest weight.
- If a tree results, it must be a maximum spanning tree.
- If not, there must be a cycle.
  - Identify the cycle and contract it into a single vertex.
  - Recalculate edge weights going into and out of the cycle.

Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing Week 6, Lec 2 16

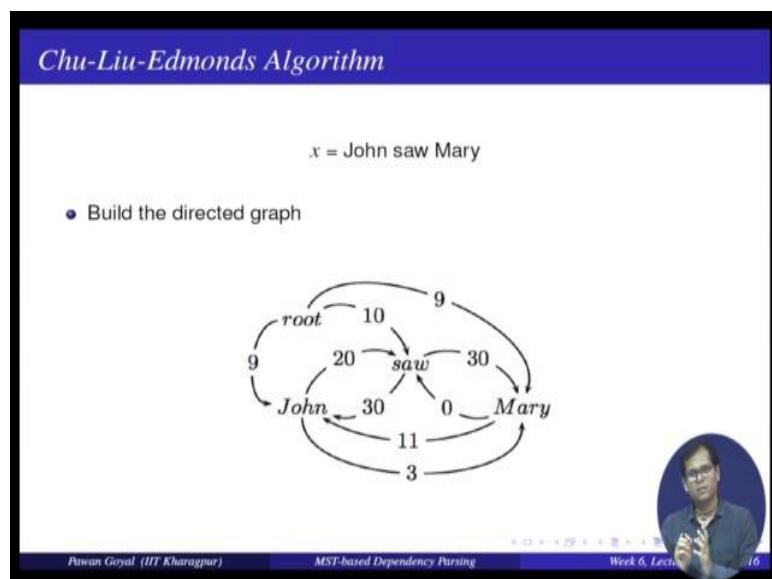
So now from a sentence I can construct this initial graph. And assume that I have some method of finding out what are the weights of my edges. We will discuss this problem in the next lecture how do I find the weights, but for now let us assume that we have some method of finding the weights of my edges. So I obtain a multi digraph now my problem is how do I find the MST of that multi digraph. And for that we use a very famous algorithm called Chu Liu Edmonds algorithm. And that is the very simple algorithm as well.

So let us see what this algorithm does. So very simple steps each vertex in the graph greedily selects the incoming edge with the highest weight. So for each vertex in the graph I have incoming edges from every other node. So what we will do in the first, what I will do in the first step? For each node I will select the incoming edge with the highest

weight. So now, I will now consider only those edges that have been selected. I will see whatever graph is resulting is it a tree or not. If it is a tree it has to be a maximum spanning tree. Because each node as related the maximum incoming edge. Now the problem is if whatever you obtain is not a tree. If this is not a tree; that means, there has to be a cycle.

Now, if this is a cycle then, there is a step that says you contract the vertices that are involved in the cycle in a single vertex. Now recalculate the incoming and outgoing edges for all those, for this new vertex that I have constructed, rerun the algorithm that is again for each vertex choose the edge with the highest incoming edge. Again see if the if whatever you have finding is a tree or not. If it is a tree you stop otherwise, you continue. And you can see that you will converge at some point. Why because at every step I either finding a maximum spanning tree. So I mean stopping or I am finding a cycle. If I am finding a cycle I will contract the vertices so; that means, it will reduce the number of vertices that I have in my tree. So it at some point I will have only one vertex. So that that is the point I will have to stop anyway. So this will converge. So now, again here are many questions like how do I compute the weight for the incoming arcs and outgoing arcs.

(Refer Slide Time: 14:28)



So let us see by an example. So I take the same sentence that we discussed in the previous method also the simple sentence John saw Mary. And I also know what is the

dependency graph that I want to obtain. Now how do I start mounds algorithm. So let us see the first thing, would be I start by taking all these 3 words as my nodes plus root as an additional node. So here we have root John saw and Mary, has 4 different nodes. Next you make an edge between every 2 nodes in the graph.

So from root you will have only outgoing edges from root you have an edge to saw to John and to Mary and for every other pair of words you have an incoming and outgoing edges. So from saw to Mary and Mary, to saw, saw to John, John to saw and so on. And there are some edge weights and we are not bothering right now and how do we achieve these edge weights. So let us see I am given the sentence, I have some way of finding the edge weights and I complete I construct this initial multi digraph.

Now, once I have this this digraph what is the next step of my algorithm? The next or the very first step says that each node in the graph chooses the edge with the maximum. So chooses the incoming edge with the maximum weight simple. So there are only 3 nodes here that are having incoming edges. So let us see what is the edge that I will choose. So saw has incoming edge of weight 10 20 and 0. So it will have to choose there is with weight 20. Similarly, John has incoming edge of weight 9 30 and 11. So it will choose the one with weight 30 and Mary has 30 - 9 and 3, so it will choose 30 and then I will remove all the other edges, so I have, I will have only 3 edges in my graph.

Is this is this does the qualifier for the maximum spanning tree or the directed spanning tree. It satisfies the first tree first 2 conditions, number of nodes will be same as the original graph and number of edges will be one less than the number of original nodes in the graph. So I will now have 3 edges, one for each node as the incoming edge, but that condition is there should not be any cycle. So let us see do we get a cycle or do we get a tree.

(Refer Slide Time: 16:55)

*Chu-Liu-Edmonds Algorithm*

- Find the highest scoring incoming arc for each vertex

*root*

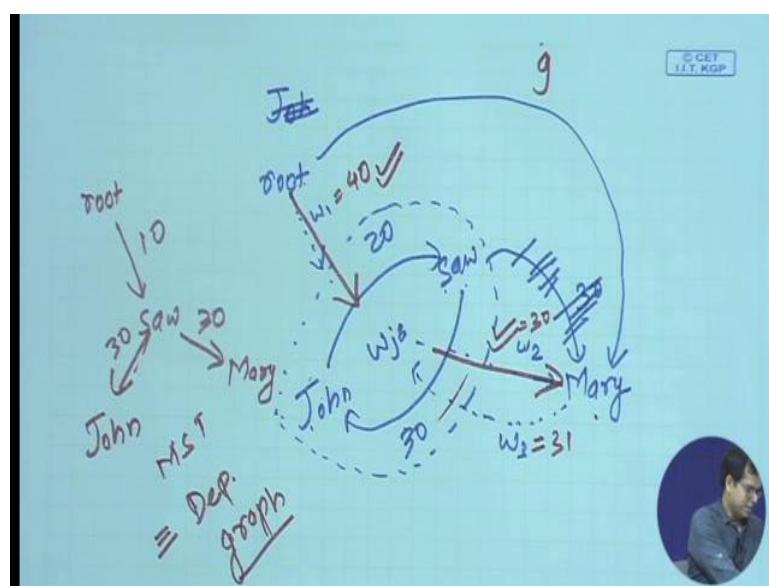
```
graph LR; John -- "20" --> saw; saw -- "30" --> Mary; John -- "30" --> Mary;
```

- If this is a tree, then we have found MST.

Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing Week 6, Lecture 4 11 / 16

So if I do that I will get a graph like this. Because saw selects this edge, John select this edge and Mary selects this edge. So what are you seeing here if there is a cycle? So I find a cycle here from John to saw and saw to John so; that means, so I am not done yet I have to now continue my algorithm. So what was the next step? Whenever I find a cycle I contract that I make a single vertex.

(Refer Slide Time: 17:29)

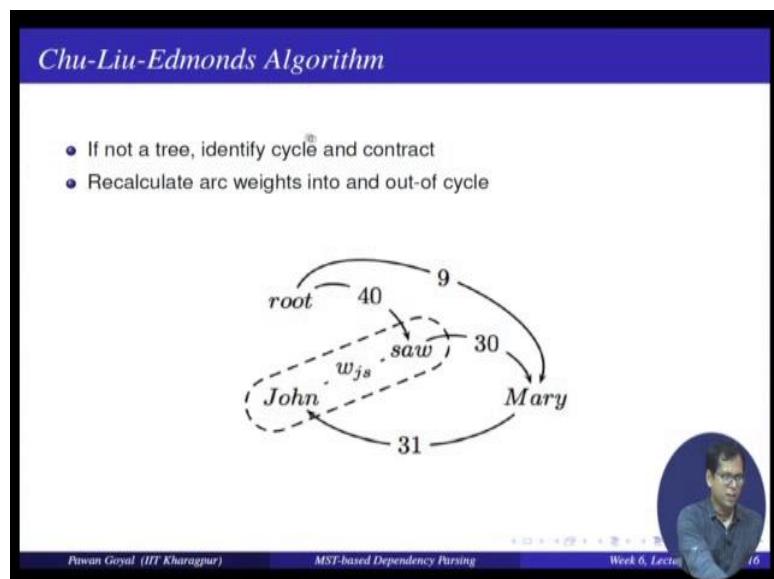


So now I have a root node, then John saw Mary. So what do I have found till, now 20 30 30. This is not a tree it is a cycle. So what is the next step? I will take it; I will contract it

into a single vertex. So there is a word j and s let us let us call it w<sub>js</sub> this is a single vertex now.

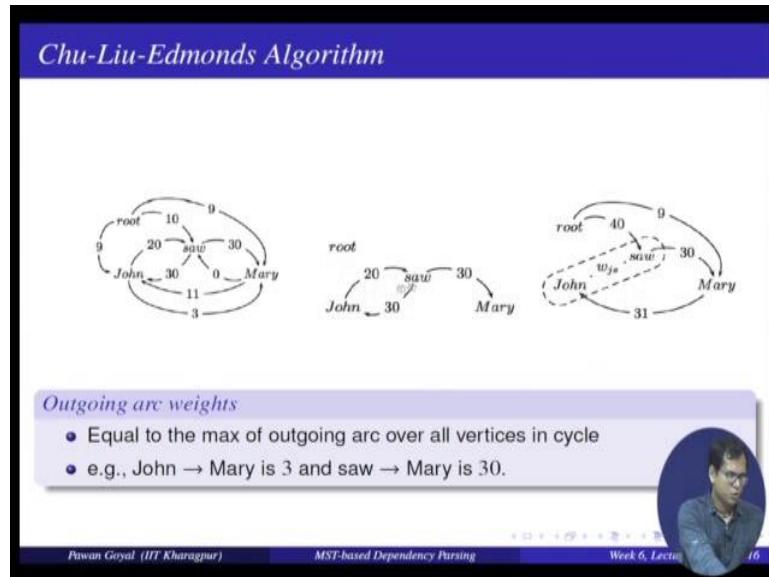
And I will have to repeat my algorithm. For repeating the algorithm, I should know to this vertex or to this new vertex w<sub>js</sub>, what are the incoming and outgoing edges from all the nodes. So there are 2 nodes now, so root and Mary. So from root it will have an incoming edge from Mary it will have incoming edge as well as outgoing edge. So question is how do I compute the incoming and outgoing weights from this vertex and for that we have some very different tools. How do I compute the outgoing edge weights and how do I compute incoming edge weights? So let us see how many I have to compute, I have to compute this weight 1 I have to compute. So now, let us forgot about this edge I have to compute weight 2 and I have to compute weight 3 yes. And root to Mary will be already there that will not change, yes. So I have now 1 2 3 edges 3 vertices only and I have to compute these edge weights. So how do I do that?

(Refer Slide Time: 19:48)



So if not a tree, you have to identify the cycle and contract that and you have to recalculate the arc weights into and out of the cycle. And what is the algorithm for doing that.

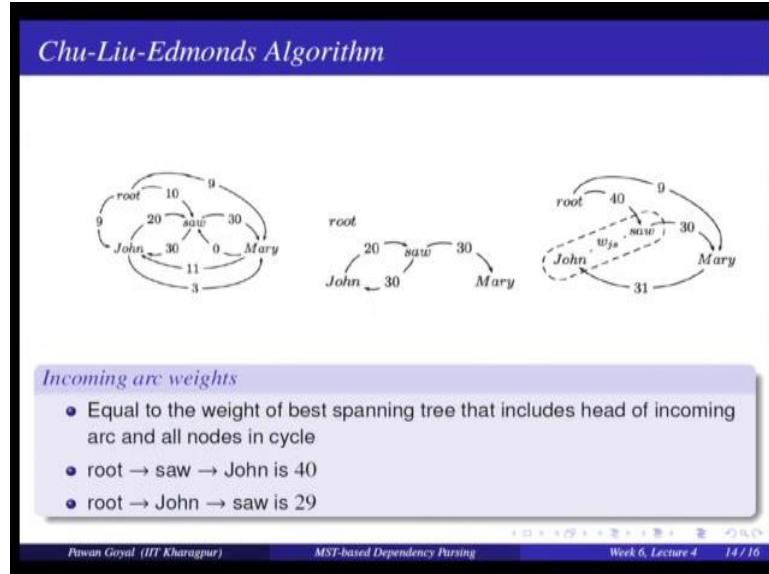
(Refer Slide Time: 19:58)



So algorithm is that for outgoing arc weights. You take the arc weights that are equal to max of outgoing arcs over all vertices in the cycle, what do I mean by this. So here is my contracted vertex John, and saw I want to compute the outgoing arc weight from this vertex to Mary yes. So this will be equal to the max of outgoing arcs from John to Mary, and saw to Mary. Which of these gives me the maximum will be final weight.

So here let us see. From John to Mary and saw to Mary; so John to Mary I have a weight of 3. And saw to Mary I have a weight of 30. So among these highest is 30. So I will pick the one with 30. And this is the final graph that is being shown. So from this vertex the outgoing edge to Mary will have a weight of 30. And that the only outgoing edge for this contracted vertex this is fine. Now how do I find out the weight for the incoming arcs? So incoming arcs are from root as well as from Mary yes, so from incoming arc the algorithm is different I do not choose the now. So I have to find out from root to this vertex. So I do not directly choose the one that is having the maximum. So what I do? I have to also consider this the edges inside the tree.

(Refer Slide Time: 21:44)



So what I will do? I will take the incoming arc weights as equal to the weights of the best spanning tree that includes the head of the incoming arc and all the nodes in my cycle. So what do I mean by that.

So let us take from root from root. There are 2 different ways of defining incoming arc weights, either it has to be root to saw and saw to John or root to john, John to saw. These are the 2 possible trees. So what I will do, I will find out the weight for individual trees and take which one is having the maximum weight. So let us say root to saw, saw to john. This is 10 plus 30 40 root to john, John to saw, this is 9 plus 20, 29. So I will take the weight as 40 and that is the edge weight from root to saw. This is having the weight of 40.

Now, can you compute the weight from, Mary to this vertex again I will see the 2 different ways Mary to saw, saw to john. So this is 0 plus 30. Choose this 30, second is Mary to John, John to saw and that is 11 and 20 31. So I have 2 ways 30 and 31. So I will choose one with 31. So it is starting from Mary to John and John to saw. So now, once I have computed all these weights I have now got the new tree on which I have to compute my algorithm. So here what did you find? This weight is equal to 40 this weight we found to 30 and this weight we found as 31 and this first (Refer Time: 23:38).

So now, once I have found these weights, what is the next step? I will have to see if each vertex now selects the incoming arc with the highest weight do we obtain a tree. So let us

see, for this vertex incoming arcs are 40 and 31. So it will choose this arc for this vertex 9 and 30. So we choose this arc. So what is my, what is the graph that I am seeing. So let me color only the edges by this. So one edge is here and the second edge is here yes. So do you see it is a tree? It is a tree and this is directed spanning tree.

So am I done? So as one point I am done because there is no cycle. So I do not have to repeat my algorithm, but I still have not the optimal dependency graph. Why? Because I am now stuck with an edge from root to this contracted vertex and I do not know what happens inside. And I know there is an outgoing edge from this contracted vertex to Mary again I want to know what happened inside. So how do I construct my full graph and for that you have to go back to how you actually constructed these connections for this contracted vertex.

So what you will do? So the incoming edge from root to John and saw and the outgoing edge from John saw to Mary. So let us see how did we compute the outgoing edge. We said the outgoing edge should be the maximum of both the outgoing edges. So from John to Mary and from saw, saw to Mary and where did these come from this came from saw. So I will now see here this edge from saw to Mary that may write it 30. So this is done. So this outgoing edge is from saw not from john. That is how we found it out.

Now it is about the incoming edge. This is an incoming edge could have been from saw then John or John then saw. In our algorithm where did it come from? 40, so it came from root to saw and saw to John yes. So I have to construct now root to saw, saw to John. So root to saw and saw to John this was 10 and this was 30. And that is my final MST that is my dependency graph. So this is the algorithm.

(Refer Slide Time: 26:57)

*Chu-Liu-Edmonds Algorithm*

Calling the algorithm again on the contracted graph:

- This is a tree and the MST for the contracted graph
- Go back up the recursive call and reconstruct final graph

Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing Week 6, Lecture 16

So let us see again here. So you are at this step where you have this contracted vertex and you found out the maximum incoming edge for each vertex and you obtain a tree. So now, you have to go back after your recursive call and we construct the original graph. And there you have to find out where is this outgoing edge coming from is it coming from saw or John, and you found it is coming from saw incoming edge is it coming for this tree, or that is root to saw, saw to John or root to John, John to saw and that you can again find out where did you find this 40, it was from root to saw, saw to John.

(Refer Slide Time: 27:34)

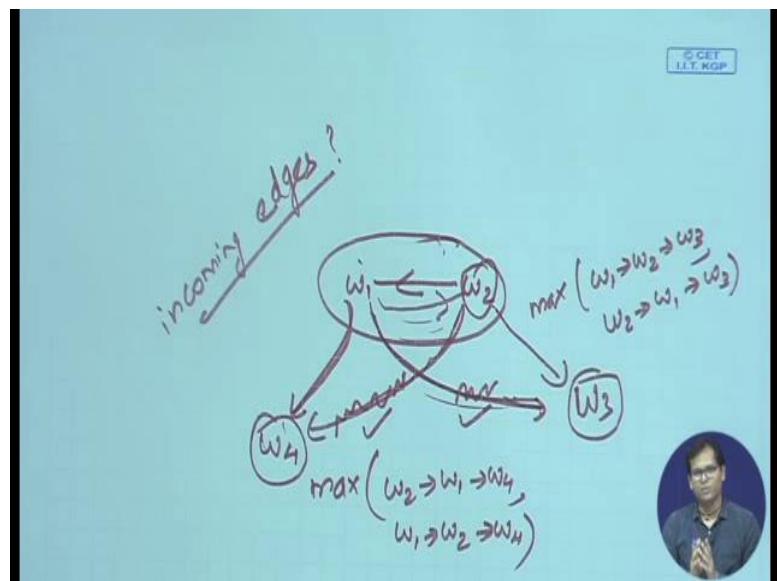
*Chu-Liu-Edmonds Algorithm*

- The edge from  $w_{js}$  to  $Mary$  was from  $saw$
- The edge from  $root$  to  $w_{js}$  represented a tree from  $root$  to  $saw$  to  $John$ .

Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing Week 6, Lecture 16

So yes the edge from w<sub>js</sub> to Mary was from saw. So we construct that the edge from root to w<sub>js</sub>, represents a tree from root to saw, saw to John and that is your dependency graph. So now, there might be a question in your mind that in my algorithm I have different ways of computing incoming arc weights and as well as outgoing arc weights they are not the same. So we need them to be different. So that we can also account for the weight for the connection inside the tree, but the question is can I reverse this, that is can I say that the outgoing weight should be the maximum of the directed spanning tree and the incoming weight should be the maximum among all the possible incoming weights. So I would like you to think about it, but I can give you basic intuition that why this is the case.

(Refer Slide Time: 28:40)



So let us see. So suppose I have a contracted vertex that has nodes word 1 and word 2. So right now what we are doing? We are saying the outgoing weight is max of to a vertex like w 3 it is a max of these 2. So we take the max here. Suppose there is another word x w 4. I will do the same the outgoing weight from here to w4 is the max of these 2.

Now when I have taken the max here and I am done my with my call and everything, each vertex will choose only one incoming edge weight. So w 3 will choose either this or this w4 will choose either this or this or anything else. So that means, I will know for sure whether this there is connection from w 2 to w 3 or w 1 to w 3 or there is no connection. And whatever I obtain even if I say that from w1 suppose my finally, I find

out a situation like this that is from w 1 there are both the connections are there this is valid. From a because from a single node you can have 2 different outgoing edges, this is fine. Or if w 1 and w 2 that is also, in no case there will be a there will be 2 incoming edges for the same node this is not allowed by the algorithm itself.

Now, suppose I change my algorithm, and say that the outgoing edge is coming from the directed spanning tree. So that will be the situation where I will have to see. So this edge is the max of w 1 to w 2 to w 3 and the other one that is w 2 to w 1 to w3, yes. Similarly, this will be the max of w 2 to w 1 to w 4 and w 1 to w 2 to w 4. Now what might happened that in your final tree, that you obtain you find out this edge yes and this edge.

Now, you have to go back and construct your tree. So what you will see there, you will see that this edge is coming from by following this connection, and this is coming by following this connection. So you will end up by saying there is a cycle inside. And this will not terminate again you can also try to see what will happen in the case of. So this is only for the outgoing edges. So you see there might be a case where you can end up finding a cycle or may not be able to say which of these 2 is the correct sign between the edges. So this case might happen this will never arise in the way we have define the algorithm.

Similarly, you can try for the incoming edges. Would you end up with the situation where you are either violating some principle of the dependency graph or where you are getting the cycle and you are not able to converge? This is something I will say that you try to take both the cases and see whether you are finding why the algorithm proceeds in this way of for finding the incoming and outgoing edge weights.

So we discuss this algorithm of Chu Liu Edmonds and how do we use that for finding maximum spanning tree. So what we did not cover is how do we find the edge weights. And that is where the learning algorithm comes in that how do we find the edge weight from my dependency graph. And that is what we will see in the next lecture.

Thank you.

**Natural Language Processing**  
PProf. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 31**  
**MST - Based Dependency Parsing: Learning**

Welcome to the final lecture of this week. So, in the last lecture, we had talked about what is the approach of using maximum spanning tree for dependency parsing. So, there we only covered that if we have a sentence and we have some way of constructing the original graph multi diagram where there all possible connection we can usually (Refer Time: 00:39) algorithm to find out the MST, in that is what we covered, now important point is how do we find out the aspects and that is what we will be covering here. So, how do we treated as a learning problem where we given some data where we know these are sentences and they are corresponding dependency graph, how do we use that to establish the aspects for a new sentence and you understand now that once you can do that part, rest part is already defined by the algorithm. So, your main task is how do you effectively construct the aspects?

Now, just one thing what do I mean by the aspect? Aspect is something that will depend on the 2 nodes that you are connecting and what is the relation with which we are connecting them? So, this has to be some sort of a feature representation vector over the nodes and the label and you have to define what are your features and once you define for any possible connection you can find out this vector; find this vector is a separate task, it does not require any learning, the way we will pick a learning involved, each features will also have a weight assigned that will help you to determine the aspects. So, feature vectors when you multiple the weight, you will get the aspects and this weight vector for the feature is something you can learn from your data. So, that is what weight vector with this feature set help me to choose the optimal or the best a spanning tree that corresponds to dependency graph that I already have in my data if I have some weights that are not allowing me to choose the best graph I should a bit my weight and this is the idea that we will see in this lecture.

(Refer Slide Time: 02:34)

### Arc weights as linear classifiers

$$w_{ij}^k = w \cdot f(i, j, k)$$

- Arc weights are a linear combination of features of the arc  $f(i, j, k)$  and a corresponding weight vector  $w$
- What arc features?

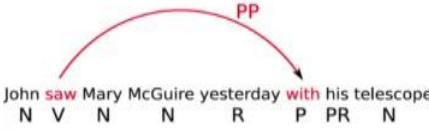


Piyan Goyal (IIT Kharagpur) MST-based Dependency Parsing: Learning Week 6, Lec 10

We will be talking about the learning part here. So, the first thing that we should see here is the arc weights that we are defining can be written as some feature vector over  $i j k$  yes the node I, the node j and the relation k and a weight vector and I multiply by weighted feature I get the arc weight.  $w_{ij}^k = w \cdot f(i, j, k)$  Now we have to see how to define this feature weight features this is again analogues somehow to what we did in some of the previous classes and how do we learn this weight vector. So, here arc weights are nothing but linear combinations of features of the arc and a corresponding weight vector. Now what we are different arc features you can take. So, it will be dependent on again what are the words I am connecting

(Refer Slide Time: 03:26)

### Arc Features $f(i, j, k)$



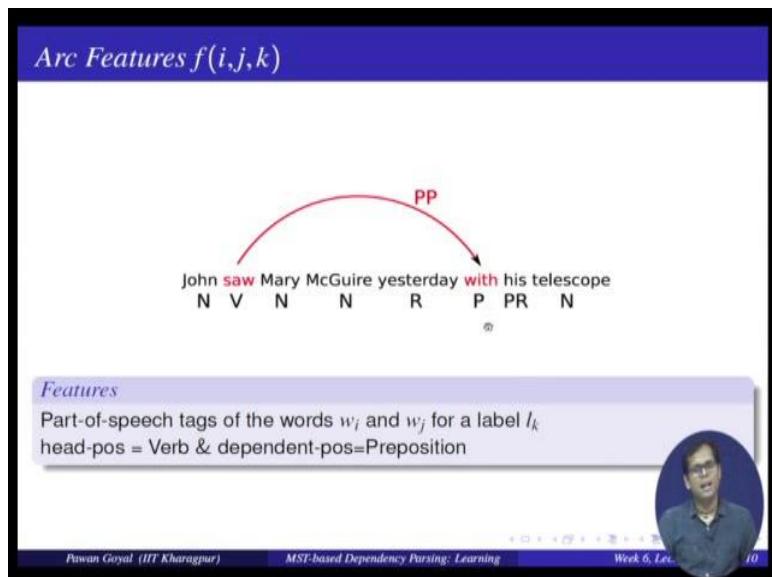
Features  
Identities of the words  $w_i$  and  $w_j$  for a label  $l_k$   
head = saw & dependent=with



Piyan Goyal (IIT Kharagpur) MST-based Dependency Parsing: Learning Week 6, Lec 10

Let us take a sentence like John saw Mary McGuire yesterday with his telescope and I want to find out the head weights between first saw and with the relation pp. So, this is my i, this is my j and this is the relation PP. So, here by the multi diagram arc means for k, you can have many variations you can have PP, you can have N subject, direct object, indirect object and non modifier etcetera, we have the various possible variations. So, what would this feature dependent on. So, feature would dependent on probably what are my words themselves they are some of the most important features if my word is saw and my word first word is saw, second word is with and the relation is whatever PP here. So, head is saw and dependent is with that is taken in my one primarily feature. So, every case I will have this feature I will have an answer 1 or 0.

(Refer Slide Time: 04:31)



Then I can have a feature composed from the part of speech tags of these words. So, I also know what are part of speech tags and. So, the feature can be like what is the part of speech for the head part. So, head part of speech is verb and what is the part of speech of dependent word that is preposition and that would be one of the most important features that I can use because many a times the relations are defined between various grammatical categories like for subject it will be between verb and a noun. So, if you know that the first the head word is verb and the dependent is noun, there is a good chance they may have a subject relation although there are other relation possible. So, part of speech is one very important feature.

(Refer Slide Time: 05:16)

### Arc Features $f(i,j,k)$

Features  
Part-of-speech of words surrounding and between  $w_i$  and  $w_j$

- inbetween-pos = Noun
- inbetween-pos = Adverb
- dependent-pos-right = Pronoun
- head-pos-left=Noun

Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing: Learning Week 6, Lecture 5 5/10

Then what else? You can also use the part of speech of the words that are coming around, this head and dependent word in between like in between part of speech is noun; that means, there is a noun that occurs in between these 2 words yes in between part of speech is adverb there is adverb that is occurring dependent pos-right is pronoun for the dependent the right hand side the part of speech is pronoun and head pos left is noun to the head to the left of the head the part of speech is noun. So, like that you can have again have many different features on the neighboring words and the words in between. So, you can use the identity of the words or the part of speech.

(Refer Slide Time: 05:58)

### Arc Features $f(i,j,k)$

Features  
Number of words between  $w_i$  and  $w_j$ , and their orientation

- arc-distance = 3
- arc-direction = right

Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing: Learning Week 6, Lecture 5 6/10

And then you might also want to use the orientation whether the head is towards light left or towards the right so that can be captured by arc direction this is to the right. So, arc direction is right you can also see the distance how many words are intervening between the head and dependent.

So, in this case there are 3 words that are intervening. So, like that you can you can take various different question or various different characteristic of head and dependent surrounding words and the labels and construct all your features. So, this gives you the set of arc features  $f(i, j, k)$  and now given any possible arc in your graph you can construct this feature vector yes it will be answer to all these questions is the arc distance 3 or not. So, answer will be one or 0. So, like that you can construct this feature vector, but what is your aspect that is the feature vector multiplied by the weight vector. So, next thing is how do we use this weight vector or how do we obtain this weight vector. So, as such you can use any features over the arc  $i, j, k$  and input  $x$ .

(Refer Slide Time: 07:16)

*Arc Features  $f(i, j, k)$*

*Features*

- Combinations  
head-pos=Verb & dependent-pos=Preposition & arc-label=PP  
head-pos=Verb & dependent=with & arc-distance=3
- No limit : any feature over arc  $(i, j, k)$  or input  $x$

Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing: Learning Week 6, Lecture 5 7/10

(Refer Slide Time: 07:20)

The slide has a blue header bar with the text "Learning the parameters". The main content area contains the following text and equations:

- Re-write the inference problem

$$\begin{aligned} G &= \arg \max_{G \in T(G_x)} \sum_{(i,j,k) \in G} w_{ij}^k \\ &= \arg \max_{G \in T(G_x)} w \cdot \sum_{(i,j,k) \in G} f(i,j,k) \\ &= \arg \max_{G \in T(G_x)} w \cdot f(G) \end{aligned}$$

At the bottom of the slide, there is a navigation bar with icons for back, forward, and search, and text indicating the source is "Piyan Goyal (IIT Kharagpur)" and the topic is "MST-based Dependency Parsing: Learning". It also shows "Week 6, Lecture 5" and "8 / 10".

$$G = \operatorname{argmax} \sum_{(i,j,k) \in G} w_{ij}^k$$

Now, how do you obtain my weights? For that I have to define, what is my inference problem? So, this we have already seen for a graph the weight of a graph will be defined by summation over the weights of all its edges and the MST problem is find out the graph with the maximum weight among all the possible spanning trees that I can obtain from the starting graph take the one that is having the maximum weight now this  $w_{ijk}$  can be written as  $w$  times the feature vector  $f_{ijk}$  and  $w$  can go outside of this equation.

So, this will be in other words  $w$  times summation over all the feature vectors and take the arc max over that and suppose I call this as the feature of my graph the feature vector for my graph is nothing, but the summation over feature vectors of all the edges and this is my problem  $\operatorname{arcmax}_w w \cdot f(G)$  over all the possible graphs or directed spanning tree  $g$  that I can construct from my initial graph now how do use this inference problem to learn my bits.

(Refer Slide Time: 08:47)

### Inference-based Learning

```

Training data:  $T = \{(x_t, G_t)\}_{t=1}^{|T|}$ 
1.  $w^{(0)} = 0; i = 0$ 
2. for  $n : 1..N$ 
3.   for  $t : 1..|T|$ 
4.     Let  $G' = \text{argmax}_{G'} w^{(i)} f(G')$ 
5.     if  $G' \neq G_t$ 
6.        $w^{(i+1)} = w^{(i)} + f(G_t) - f(G')$ 
7.        $i = i + 1$ 
8.   return  $w^i$ 

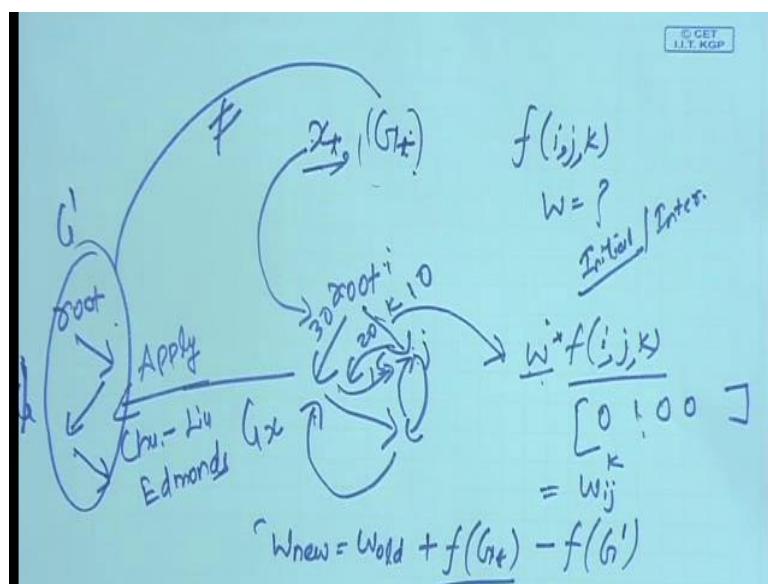
```



Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing: Learning Week 6, Lec 10

This will be my learning criteria and this will be very similar to what we did in the previous example that the previous method also. So, what is the idea? You have a gold standard training data. So, you are given a training data of a sentence and its gold standard dependency graph. So, there are some capital T sentence that is (Refer Time: 09:12). Now start with some initial weights they can be 0 or they can be some arbitrary weights and let us start with the first iteration now I am iterating over all the sentences some capital n number of times like we did in the previous case in each iteration I go to each of the individual tree, so, 1 to up to capital T.

(Refer Slide Time: 09:55)



Each sentence instance now what do I do. So, I am in the learning part, I do not know, what are my optimal weights? So, how does learning work. So, I am given a sentence  $x_t$  and its dependency graph  $G_t$ , I am already given this I know what are my features yes I know what are my features I do not know what are my weights. So, how do I proceed?

Let us take a particular instance for my learning. So, what will happen? I am given this sentence  $x_t$ . So, I will apply my algorithm as if I am doing it at run time. So, I will make the root node, I will make all the possible connections, somehow, we can have any possible connection that is not matter. So, now, you start by making all the possible connections that we discussed, this is my initial graph. You can (Refer Time: 10:57) that graph  $G_x$  that has all the possible connections this also all the possible connections.

Now, at this point how will you proceed for proceeding you need to know, what are the edge weights here? So, now, how do you obtain the edge weights? So, this edge; it will be nothing but  $W$  times  $f_{ijk}$ ,  $i$  is the  $i$ th word here,  $j$  is the  $j$ th word here and  $k$  is the relation, now how do you obtain  $f_{ijk}$ ? This already given to you, so you know, what are the features? So, features will be some question like  $i$ th words is root and  $j$ th word is something like saw it will be either 1 or 0, yes. So, your feature vector will be some 0 some ones and this might be different for different edges.

Now, what else you do not know your weight, but you will have initial weights or some intermediate weights you will use those weights multiply by the feature vector and you will get some number and that is your  $w_{ijk}$  yes and you will put back this number it can be say 10 or 30, whatever suppose you get 10, 30, 20, 15 and so on, you will filling all the values using the same method. Now once you have filled in all these values what will be the next step. So, next step should be apply Chu Liu Edmond's algorithm yes and obtained a dependency graph and in other words obtained a MST. So, suppose you obtain a graph like root here. So, there are 3 words. So, something like this suppose let us take a simple case and like this, this a one of the dependent one way of writing dependency graph and suppose this is what you obtain from your algorithm.

Now, where is the learning here learning is if the dependency graph that you obtain is if this is not the same as your gold standard dependency graph; that means, your weights are not correct at this point you might have to modify your weights and how do you modify your weights very simple like you did in the previous method also you would say  $w_{\text{new}}$  or  $w_{\text{old}}$  plus you want

to go towards the actual graphs and away from them. So, call it  $G$  prime that you obtain from by your algorithm. So, call it  $G$  prime that you obtain by your algorithm;  $G_t$  is the gold standard graph.

You want to go in this direction and away from here. So, it will be simply feature of  $G_t$  minus feature of  $G$  prime and what is  $F_{G_t}$ ? This is nothing but the summation over features of all the edges of  $G_t$ , similarly here summation over features of all the edges of my  $G$  prime and that will give you your new weight vector of course, there can be some learning weights and all, but this is just to give you the intuition and that is all you will have a new weight. Again you will start with this new weight for a new sentence see whether you obtaining the  $G_t$  like show graph by this algorithm if not you will keep on updating your weights and this will sometime at some point converge and some of one of your iteration you will have your weights roughly stabilized and that is where you stop. So, now, let us so, hope this idea is clear, let us take a simple example to further see that how do you apply this algorithm for learning the weights.

Let me just quickly go through this algorithm again. So, I am repeating it for some  $n$  number of iterations for each tree let us find out  $g$  prime as the Argmax over weight at the particular instance and  $f_{G_t}$ . So, that is I apply my Chu Liu Edmond's algorithm find out the tree with the maximum weight, if  $G$  prime is not the same as my gold standard graph; that means, I have to update my parameters and how do I do that at this I plus one th time my weights are initial weights plus  $F_{G_t}$  minus  $F_{G\text{ prime}}$ , this I am doing only if I am not obtaining the correct graph, I continue further and at some point it will converge and that is where I will return the weights and I will assume at this point I have the ideal weights. So, that whenever you give me a new sentence I can use these weights to compute all the arc weights yes that arc weights are nothing, but weights multiplied by the feature vector and then I can I can apply Chu Liu Edmond's algorithm to compute the dependency graph.

(Refer Slide Time: 16:28)

**Example**

Suppose you are training MST Parser for dependency and the sentence, "John saw Mary" occurs in the training set. Also, for simplicity, assume that there is only one dependency relation, "rel". Thus, for every arc from word  $w_i$  to  $w_j$ , your features may be simplified to depend only on words  $w_i$  and  $w_j$  and not on the relation label.

Below is the set of features

- $f_1$ :  $\text{pos}(w_i) = \text{Noun}$  and  $\text{pos}(w_j) = \text{Noun}$
- $f_2$ :  $\text{pos}(w_i) = \text{Verb}$  and  $\text{pos}(w_j) = \text{Noun}$
- $f_3$ :  $w_i = \text{Root}$  and  $\text{pos}(w_j) = \text{Verb}$
- $f_4$ :  $w_i = \text{Root}$  and  $\text{pos}(w_j) = \text{Noun}$
- $f_5$ :  $w_i = \text{Root}$  and  $w_j$  occurs at the end of sentence
- $f_6$ :  $w_i$  occurs before  $w_j$  in the sentence
- $f_7$ :  $\text{pos}(w_i) = \text{Noun}$  and  $\text{pos}(w_j) = \text{Verb}$

The feature weights before the start of the iteration are: {3, 20, 15, 12, 1, 10, 20}.

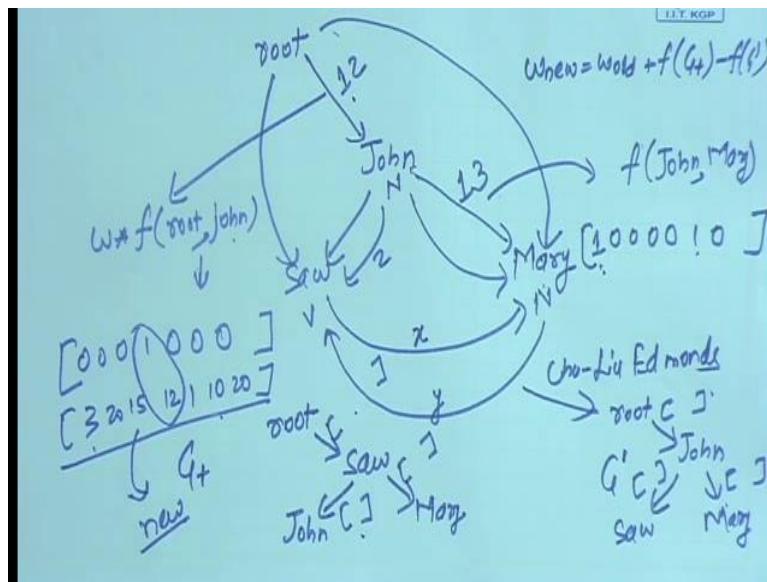
Determine the weights after an iteration over this example.

Pawan Goyal (IIT Kharagpur) MST-based Dependency Parsing: Learning Week 6, Lecture 5 10 / 10

Let us see some example for learning the weight vectors. So, we are using the same sentence like the previous case. So, we are using the sentence John saw Mary and suppose that this occurs in the training set now for simplicity we are not worrying about what are the relations between 2 words I am saying that there is only one relation rare. So, my features will only dependent on the ith word and jth word that will make things easier. So, here what I am showing? Let us take some 7 different features and some weights that have been initialized or some intermediate weights and suppose you are un counting this sentence in your training data how will you do the weight update.

Here let us see what are my 7 features defined over any word i and j for a arc from  $W_i$  to  $W_j$ . So, features are the ith word has a part of speech of noun and jth word has a part of speech of noun, ith word is part of speech of verb, jth word is part of speech of noun and so on, ith word is root, jth word is verb and for this simple sentence you already know.

(Refer Slide Time: 17:39)



You will have root John, let me write it like that saw Mary you know this is a noun, this is a verb and this is a noun and this is root. So, now, your task is suppose you are given for these 7 features the weights are 3, 20, 15, 12, 1, 10 ,20, before the start of the iteration, now you have to determine the weights after iteration over this example, now what do I mean by doing that same as I said in the last slide. So, what I will do? Now I will first find out all the possible edge weights, let us try to find some edge weights what is the edge weight from root to John. So, what is this edge weight? This is weight vector multiplied by the feature vector between this root and John this might  $W_i$ , this might be  $W_j$ , now what is this  $F$  root to John? This will be a vector of size 7, each value to be 1 and 0 and this will dependent on the answer of my question.

Let us see what are my features word?  $W_i$  has a part of speech of noun now  $W_i$  is a root. So, it does not have any part of speech this will be 0  $w_i$  has part of speech of verb again 0,  $W_i$  is root, let's see the next 1  $w_j$  is verb. So, here  $w_j$  is John that is noun. So, this is also 0 root and  $j$  is noun this will be 1 root and  $w_j$  occurs at the end of the sentence know John occurs in the John occurs in the start of the sentence this is also 0,  $W_i$  occurs before  $w_j$  in the sentence roots does not occurs in the sentence. So, this will be 0 also  $W_i$  is the part of speech of noun. So, it is not true for root. So, we see out of 7 features only 4th one is one everything has a 0. So, the feature vector here is 0001000, now how do I get the weight the weight vector is already given to me that is 3, 20, 15, 12, 1, 10, 20. So, I multiple this weight vectors to the feature vector and I obtain the weight of this arc that is 12.

Similarly, I compute the weight of the arc of from root to saw. So, let us take some other case like John to Mary, how do I compute the arc weight? Again this is let us find out the feature for John to Mary. So, first 1, first question again W i is the part of speech of noun w j is noun, now John and Mary both are nouns. So, this should be 1, W i is verb and w j is noun should be 0, W i root none of these. So, none of these features will be 1 because all the required (Refer Time: 21:12) to be root. So, the all 3 are 0. So, 2, 3, 4, 5 are 0 6, W i occurs before w j. So, John occurs before Mary in the sentence there should be 1 7th feature, W i had part of speech of noun, w j is verb that is 0. So, only 1 and 6 are 1, everything else are 0. So, it will be 1000010 and multiplied with the weight vector. So, similarly first element and 6th element; 3 plus 10, so 13, so here the weight will be 13.

Like that you have to find the feature vector for each edge multiplied by the weight vector and obtain these values you will said get some values here x y z etcetera now once you obtain all these values what you will do apply Chu Liu Edmond's. So, once you have all the weights, you can very easily obtain Chu Liu Edmond's and you will obtain what is your dependency graph now what will happen suppose you find a graph like this root John saw Mary suppose, this is your output after applying Chu Liu Edmond's, now what you will do? So, you will; this is your G prime by applying Chu Liu Edmonds. So, your first thing you will check is that what is your G t? So, your G t is root remember you will first have connection from root to saw to John and Mary, this is my G t, are they same? They are not the same. So, I will update my weights.

And how do I update my weights (Refer Time: 23:11) should be W new is W old plus f G t minus f G prime, now what is f g t? F G t is the feature of this graph that is this feature vector plus this feature vector plus this feature vector 3 feature vectors combined and f G prime is this feature vector plus this feature vector plus this feature vector and these are all some 1s and 0s. So, weights will be nothing, but these weights plus summation over all these 3 elements of features minus all these 3 elements and that will give you the new set of weights and then you will continue with this new set of weights for the next example if you have to do.

This is in a nutshell, what we will be doing, but again I will encourage you to go through these example fully find out all the weight vectors yourself see whether you are getting the same tree as the gold standard if not how will you update your weights in this will be your next exercise. So, here we are not going to very much detail in how we are going to learn it by various learning rates and all just an intuition that how do you post this as a learning problem? So, in the last module of using (Refer Time: 24:45) parsing or transition parsing and this module of using

MST based parsing, we have seen that how we can solve a problem dependency parsing in a detailed different manner. So, we treated as some sort of algorithmic way I was starting from some initial configuration going to some final configuration that resembles a dependency graph.

And in the in between for example, how are we taking transitions. We will determine by using various training data or how are we taking the edge weights, we are learning by training data and this would help you to also think about in terms of some other problems that how can I pose them in machine learning way and I was also saying there are some other they are some other or I would say even many other methods of for doing dependency parsing. So, we will not be covering everything in this course, but now whatever we have covered will help you to take any new approach and understand that. So, this also ends our discussion on main discussion on syntax. So, we had talked about starting from word order information language models to part of speech and morphological information to various syntaxes in the sense of constituency parsing and dependency parsing.

From the next week, we will start our discussions on semantics. So, we will first see what is called different methods by which being capture the meaning; the semantics. So, we will have distribution semantics and (Refer Time: 26:19) semantics for that.

Thank you, I will see you in the next week.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 32**  
**Distributional Semantics – Introduction**

Welcome everyone to the 7th week of the course. So, with that we are now moving into the second half. So, in the first 6 weeks we had talked about basic tasks on in any language, how do we pre process the language, how do we deal about its morphology and syntax. Now we will be going to some advance topics. So, in the next 3 weeks, starting this week, we were talking about semantics, how do we capture semantics?

Again semantics is very huge topic, we will focus only on very very basic and interesting notions that can you help you build some important applications and also read some research papers in this field. Later in the last 3 weeks of the course, we will be going to various applications. So, today we are starting our discussion on semantics. So, and we will be taking about 2 notions distribution semantics and then lexical semantics. Before going into that let us see what do I mean by semantics? So, in this lecture today we will give the introduction to the topic of distributional semantics.

(Refer Slide Time: 01:27)

*Introduction*

**What is Semantics?**

**The study of meaning:** Relation between symbols and their denotata.  
John told Mary that the train moved out of the station at 3 o'clock.

Pawan Goyal (IIT Kharagpur)      Distributional Semantics - Introduction      Week 7, Lecture 1      2 / 14

As such when I say semantics, what comes to your mind? In general syntax talks about arrangement, how are that different words are arrange in the language in when while talk

about semantics, it immediately we move from arrangement to some sort of meaning domain. So, I want to find out, what is the meaning of different words, how do they combine together to form the meaning of the larger unit that can be the sentence; that is in general the main field.

If I have to give a definition, I can say that semantics is the study of meaning that is, what is the relation between symbols and what do they denote? So, here suppose I have the sentence, John told Mary that the train moved out of the station at 3 o'clock, it is a sentence in the national language contains multiple words, we know how to tokenize them and how to find the morphological tax and may be also the syntax and different representations. Now what do I mean by saying that now I want to go to semantics? In semantics, I would like to know, what all these words themselves mean separately. So, what do I mean by John told Mary, etcetera. So, these are all symbols, now what do they denote? And when they combine together in a sentence, what meaning are they putting? So, this is as I said very very vast field, lot of different research has happened and is happening, we will focus on mainly the word meaning. So, how do we say, what is the meaning of a word, what are different notions? So, in general, so when I talk about distribution semantics or any other model that we will cover in this course, we are trying to find out, how in general computers can produce such semantic representations. So, machines can interpret some sort of semantics and that is why the idea of computational comes in here.

(Refer Slide Time: 03:36)

**Computational Semantics**

**Computational Semantics**  
The study of how to automate the process of constructing and reasoning with meaning representations of natural language expressions.

*Methods in Computational Semantics generally fall in two categories:*

- **Formal Semantics:** Construction of precise mathematical models of the relations between expressions in a natural language and the world.  
 $John \text{ chases } a \text{ bat} \rightarrow \exists x [bat(x) \wedge chase(john, x)]$
- **Distributional Semantics:** The study of statistical patterns of human word usage to extract semantics.

Pawan Goyal (IIT Kharagpur)      Distributional Semantics - Introduction      Week 7, Lecture 1      3 / 14

Computational semantics in general is the study of how we can automate this process of building these semantic representations and also reasoning with them. So, when we talk about the methods, in general there are 2 different methods, one are based on formal semantics that is how do I construct various mathematical models that can tell me, what is the relation between various expressions in the language and also relate them to whatever there is there in the word.

For example, so, if you have heard about read predicate logic that is what is done in formal semantics. So, if I have a sentence like this, John chases a bat and you want to produce a mathematical structure that denotes the meaning of the sentence, so I will put them in very very logical form like I will say so, for startup, (Refer Time: 04:33) I will say there is an x where x is a bat and John chases the bat. So, John chase becomes a predicate and I will say John chases x and x have already defined as a bat and like that I will define, how do I convert my natural language expression to some sort of this logical form and then I will also have rules on how do I infer from this logical expressions? How do I reduce one expression into another 1 and then so on?

And this is again a field of formal semantics and that is something we will not cover in this course, what is something apart from formal semantics that we can see and there where the data, a lot of data that we have can help us and this is where the field of distribution semantics comes in that is can I study the statistical patterns of human word usage to extract semantics.

Can I see how humans are using different words in th language to find out what is their semantics? And this is what will be the topic of this week, what is the type of distributional semantics? How they capture that and how can I use that for certain meaningful applications? So, this field of distributional semantics is mainly built upon this hypothesis of this hypothesis of distribution, distributional hypothesis that is prevent from many many years, what is the distributional hypothesis? Let us see some famous quotes about this.

(Refer Slide Time: 06:09)

*Distributional Hypothesis: Basic Intuition*

"The meaning of a word is its use in language." (Wittgenstein, 1953)

"You know a word by the company it keeps." (Firth, 1957)

→ Word meaning (whatever it might be) is reflected in linguistic distributions.

"Words that occur in the same contexts tend to have similar meanings." (Zellig Harris, 1968)

→ Semantically similar words tend to have similar distributional patterns.

As earlier as in 1953 Wittgenstein said that the meaning of a word is its usage in language. So, that is you can know, what is the meaning of a word if you see how it is being used in the language?

Along similar lines in 1957, Firth said you know the word by the company it keeps, now what do I mean by the company of a word? What are the other words it occurs within my corpus on my language and that tell me about the word? Now going 1 step further, so here so, what these different quotes are mentioning that the word meaning whatever it might be, I do not care about what exactly is the meaning, but whatever it is, it should be reflected in the linguistic distributions that is the way the word has been used in the language that will tell me the word meaning. So, now, Zellig Harris in 1968 that gave this famous code that took this idea to one step ahead that is words that occur in the same context tend to have similar meanings.

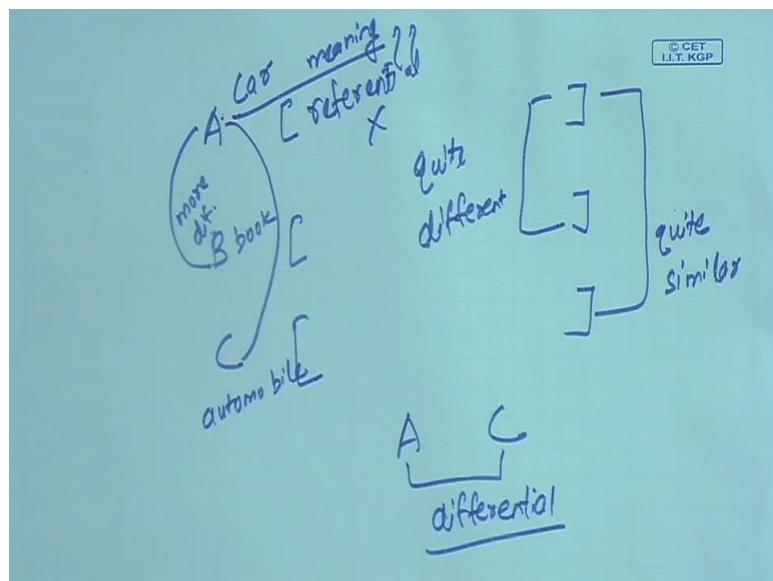
Now I can talk about 2 words having similar or different meanings, if I can somehow measure their context and capture that and I will say 2 words are having similar meanings, if the context in which they occur are very very similar. So, the idea here is 2 words, if they are semantically similar, they tend to have similar distribution patterns. So, now, that is what we will be doing in this week, how do we capture distribution patterns of words and use that to find out similarity across words.

(Refer Slide Time: 07:56)

Let us see some other quotes from Zellig Harris. So, he also said that if linguist have to deal with meaning it can only do so through distributional analysis and that he said in very early 70s and 80s and if you see the (Refer Time: 08:12) field right now, so whatever the main methods or semantics are there, they are built upon distributions. So, with the idea of; so with the recent idea of word and biddings and all and this is what we will also capture in our in this week of this course.

Another very similar sort of code from Zellig Harris from his distributional structures, so if we consider, what are morphemes A and B to be more different in meaning than A and C then we will often find that the distributions of A and B are more different than the distributions of A and C. In other words, difference in meaning correlates with difference of distribution.

(Refer Slide Time: 08:55)



What he is saying suppose I take 3 different words or morphemes A B and C and suppose there is some where I can capture the distribution. So, that is how they are occurring in the languages. So, I have distribution for A B and C, now what he is saying? If A and B, the distributions are more different than that of A and C then what you will find? So, what we are saying A and B are more different than A and C, if the distributions are more different in the meaning also, you will find they are more different. So, you can think of these as so, I take some examples like car as A and B C can be automobile, they are quite close and B can be something like a book. So, here a car and automobile are quite similar and car and book are more different. So, they are very very different. So, if this is what I see in meaning, I will also see something similar in distributions. So, what I will find? These 2 distributions are quite similar and these 2 would be quite different.

We will see in this week that how do we capture these distributions and how do we compare that, but this is the basic idea, now what is another important thing here? So, distributions that we are capturing, whatever semantics it is allowing me to handle, it is all differential not referential, now what is the difference between that 2 terms differential and referential? So, again if I would look at the same example of these 3 words; car, book and automobile, so what do I mean by saying so? The semantic sign capturing is only differential, but referential. So, I cannot say, what is the meaning of car? I am not defining the meaning of car at sort of this concept representation; there is something I am not doing. So, what I am doing? What, How similar or how different the 2 meanings are? So, I am saying A and C, how similar their

meanings are or how different their meanings are? That is why this is differential sort of understanding of semantics. So, how different or how similar 2 different meanings are I am not talk talking about some referential meaning in distributional semantics.

Now if we look at so this was the linguistic prospective is there some cognitive prospective also with distributional semantics.

(Refer Slide Time: 11:52)

*Distributional Semantics: a cognitive perspective*

**Contextual representation**  
A word's contextual representation is an abstract cognitive structure that accumulates from encounters with the word in various linguistic contexts.

**We learn new words based on contextual cues**  
He filled the **wampimuk** with the substance, passed it around and we all drunk some.  
We found a little **wampimuk** sleeping behind the tree.

Pawan Goyal (IIT Kharagpur)      Distributional Semantics - Introduction      Week 7, Lec. 14

There we have this idea that what is the representation of the word and it is said to be some sort of abstract cognitive structure that I am storing in my brain or somewhere we do not know, but that is something I gather as I keep on hearing this word or looking or finding this word in more and more context as I find this word in more and more context, I tend to build some sort of representation about this word this is the cognitive prospective and what is some sort of evidence. So, for example, when you encounter a new word that you have never heard before even if we do not know its meaning you may guess something about this word what it might be?

Let us take an example like I am saying this word wampimuk, I have never heard this word before, now I am seeing this word or hearing this word in this context, he filled the wampimuk with the substance, passed it around and we all drunk some. So, just by looking at the word that are occurring around this word and seeing what are the words might occur in place of wampimuk, I can say that this might be some sort of a container some sort of a glass or something so which can be used for filling up the substances and passing around.

As I keep on hearing a word or seeing a word in more and more context, I tend to build some sort of meaning structure about that word. Now suppose I have heard this word in a very different context like this, we find a little wampimuk sleeping behind the tree, immediately, I will have a different interpretation, I will say may be wampimuk some is some sort of a small animal. So, this is some sort of intuition behind the cognitive prospective and why we might think the meaning in terms of the distributions in the language.

(Refer Slide Time: 13:47)

*Distributional Semantic Models (DSMs)*

- Computational models that build contextual semantic representations from corpus data
- DSMs are models for semantic representations
  - ▶ The semantic content is represented by a vector
  - ▶ Vectors are obtained through the statistical analysis of the linguistic contexts of a word
- Alternative names
  - ▶ corpus-based semantics
  - ▶ statistical semantics
  - ▶ geometrical models of meaning
  - ▶ vector semantics
  - ▶ word space models

Pawan Goyal (IIT Kharagpur)      Distributional Semantics - Introduction      Week 7, Lecture 1      7 / 14

Now, to capture these distribution and then and find semantics from there, the models that are used are called distributional semantic models and also there is some term called DSM for them and what are these? So, these are various computational models that build contextual semantic representations from my corpus data. So, now, what is important here is that I am already given some sort of and if more data generally the better. So, I am given some corpus data on how different words are used in language and from there I am trying to come up with some semantic models distribution semantic model and this will capture the differential aspects of meaning among words.

DSMs are models for semantic representation and where I capture a semantic content by using a vector. So, for each word we will try to build a different vector that will capture the semantics and these vectors are obtained via the statistics analysis of the linguistic contexts of the word. Now this word occurs in various context will help to find out its vector representation and there are some alternative names for this semantics like corpus based

semantics statistical semantics geometrical models of meaning vector semantics and word space models and there might be some other prevalent names also, but we capture the same idea that can I use the distribution of the words to find out the meaning or at least capture which words are similar than other which pair of words are similar than other pair of words.

(Refer Slide Time: 15:26)

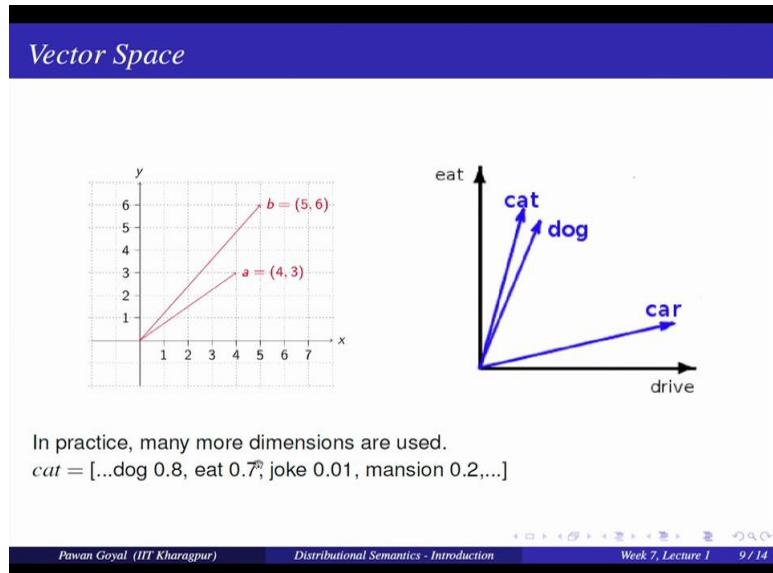
*Distributional Semantics: The general intuition*

- **Distributions** are vectors in a multidimensional semantic space, that is, objects with a magnitude and a direction.
- The **semantic space** has dimensions which correspond to possible contexts, as gathered from a given corpus.

Pawan Goyal (IIT Kharagpur)      Distributional Semantics - Introduction      Week 7, Lecture 1      8 / 14

Now again so, this distribution semantics when I talk about, what do you mean by the 2 terms distribution semantics? So, when I say distribution, they are the vectors denoting different different words and their vectors in some multi high dimensional or in low dimensional space and this space is the semantic space. So, I have distributions that are vectors denoting the words in that multidimensional semantic space and semantics space has dimensions which correspond to various possible contexts and this context can be gathered from a corpus. So, what I am doing? Every word I am denoting in this semantics space and this is my distribution semantics and this semantics space composed of various dimensions that are my context in which I am trying to represent a given word. So, let us take this an example.

(Refer Slide Time: 16:27)



When I talk about symbol vector space model, so that all of you would already be aware of, so this is my 2 dimensional plane and it is x y plane and I can denote different objects. So, here there are 2 points that I am denoting a and b by their coordinates. So, what are, so their projection in on x axis and y axis that tells me the coordinates and I can now capture how similar, how near to words 2 objects are in this space analogous to that, now let us think of a semantics space where dimensions are not x and y axis, but they are various context and can I denote all my words in those dimensions.

So, here suppose that my dimensions are 2 words eat and drive. 2 different words and I am trying to denote 3 words; cat, dog and car in these dimensions and what you are seen here? So, this might denote how often a word occurs with this word on and this word the projection will denote that. So, you see cat comes quite often with the word eat similarly dog car comes a lot often with drive, but not so much with eat. So, you see immediately when you try to put these word in this semantic space will capture some similarity that cat and dog are probably similar much more similar than cat and car and dog and car and this is the idea, can I use different context as my dimensions and represents all my words in those dimensions to give a meaning representation? In general here, I have shown you only for the dimension that corresponds to 2 words, but in general you all have to use any number of dimensions.

So, you can use any number of words that you want. So, so you might have a representation like that. So, I can say cat is a any other object or a word that has a weight of point eight in

the dimension of dog 0.7 dimension of eat 0.01 dimension of joke. So, depending on how often the word cat occurs with all these words and this I can do for different different words and then I can capture how similar they are and this is a very very powerful technique although its looks very simple it works quite well in many many applications that that we will see.

(Refer Slide Time: 18:49)

The slide has a blue header bar with the text "Word Space". Below it is a white content area. In the top left of the content area, there is a light purple box labeled "Small Dataset". Inside this box are six sentences:

- An automobile is a wheeled motor vehicle used for transporting passengers .
- A car is a form of transport , usually with four wheels and the capacity to carry around five passengers .
- Transport for the London games is limited , with spectators strongly advised to avoid the use of cars .
- The London 2012 soccer tournament began yesterday , with plenty of goals in the opening matches .
- Giggs scored the first goal of the football tournament at Wembley , North London .
- Bellamy was largely a passenger in the football match , playing no part in either goal .

Below the dataset, there is a section with two lines of text:

Target words: <automobile, car, soccer, football>  
Term vocabulary: <wheel, transport, passenger, tournament, London, goal, match>

At the bottom of the slide, there is a dark footer bar with the following text and icons:

Pawan Goyal (IIT Kharagpur)      Distributional Semantics - Introduction      Week 7, Lecture 1      10 / 14

Now let us take an example and how do we start constructing this a vector space or word space model and try to see can we compute the similarity across words. So, here I am given a small data set that has 6 different sentences in. So, like an automobile is a wheeled motor vehicle used for transporting passengers and so on, there are 6 different sentences are given and I have to build a vector space model or a distributional semantic model now to build the model, I need to start with what are the words I want to represent and these are my target words. So, here I want to represent 4 word automobile car soccer and football. So, now, once I know, what are the words I want to represent? The next question I need to ask is I need to represent in what dimensions? So, this becomes my context, here also denoted by term vocabulary what are the dimensions in which I will be denoting all these 4 words. So, here I have 1, 2, 3, 4, 5, 6, 7, dimensions.

(Refer Slide Time: 19:54)

*Constructing Word spaces*

Informal algorithm for constructing word spaces

- Pick the words you are interested in: **target words**
- Define a **context window**, number of words surrounding target word
  - ▶ The context can in general be defined in terms of documents, paragraphs or sentences.
- Count number of times the target word co-occurs with the context words: **co-occurrence matrix**
- Build vectors out of (a function of) these co-occurrence counts

Pawan Goyal (IIT Kharagpur)      Distributional Semantics - Introduction      Week 7, Lecture 1      11 / 14

Now so what will be the informal algorithm for constructing the distributional semantic space or the word space, you start by picking up the words that you are interested in; that means, the words for which you want to find the distributions and these are called your target words. So, here I start with 4 target words then you define, what are the context words also? So, here you found 7 context words, next question would be when do I say that this word occurs with another word should I say that when it occurs in the same sentence same paragraph same document or within plus minus 2 words when do I say that. So, that is called the context window.

So, I can define a context window to be maybe 5 words around the word or it can be a sentence paragraph and whatever word. So, I need to define a context window that is how many words should I consider surrounding a target word and it can be defined in terms of a document paragraph sentences and so on.

Now, a simple method would be once you defined the context window you know, what are your target words what are your context words? So, when you are defining a distributions of the target words find out how often they occur with the various context words within the context window and just write down the number this occurs with this word in this context window 2 times 3 times 0 times and so on and you define the vectors for different target words and this is this can also be called as your co occurrence matrix between the target

words and the context words now once you have built the co occurrence matrix you can think of each row of this matrix as your vector for that denotes this individual word.

(Refer Slide Time: 21:48)

Computing similarity							
	wheel	transport	passenger	tournament	London	goal	match
automobile	1	1	1	0	0	0	0
car	1	2	1	0	1	0	0
soccer	0	0	0	1	1	1	1
football	0	0	1	1	1	2	1

*Using simple vector product*

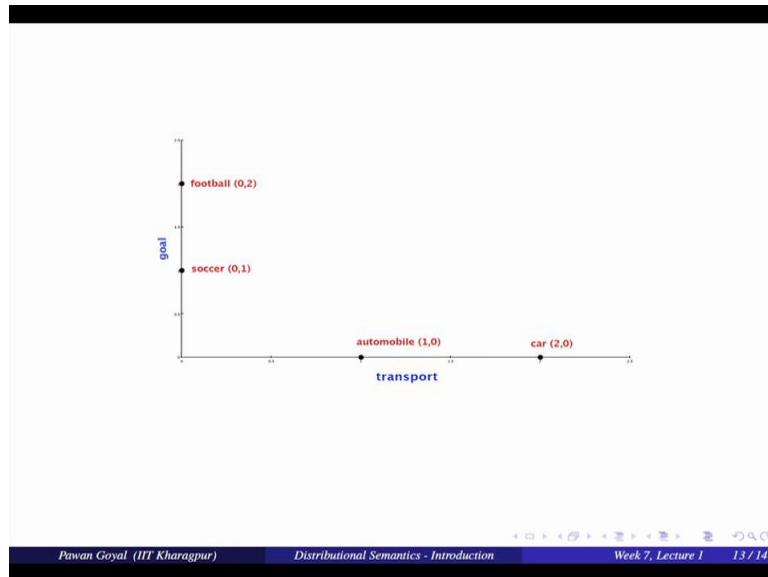
automobile . car = 4	car . soccer = 1
automobile . soccer = 0	car . football = 2
automobile . football = 1	soccer . football = 5

Pawan Goyal (IIT Kharagpur) | Distributional Semantics - Introduction | Week 7, Lecture 1 | 14 / 14

If we try to apply this algorithm on the previous example that we have seen, so what we will have? We will have 4 words as the target words, 7 word as my context words and I will have a matrix of dimension 4 cross 7 and each element will denote how often this target word occurred with this context word within my context window.

Let us see the whole sentence here in the context window. So, you can try doing that with the example that that we had seen and try to find out what are the different number of times each individual word occurs with another word and you will find something like that. So, this you can call as your co occurrence matrix or distribution matrix composed of targets and contexts. So, what do we see here? Automobile occurs with wheel, transport, passenger, football, occurs with passenger, tournament, London, goal and match and similarly for car and soccer. So, now, this is my representation of target words in this semantic space.

(Refer Slide Time: 22:53)



Now, suppose I take only 2 dimensions of this semantic space. So, here I had 1, 2, 3, 4, 5, 6, 7, dimensions; suppose I see only 2 dimensions. So, here I am seeing goal and transport and I am projecting all the 4 words in this dimensions. So, what representation I will see? So, I will see that soccer occurs with goal and goal not be transport, football occurs with goal not be transport, automobile occurs with transport not with goal and car occurs with transport not with goal. Immediately you can see some separation that is soccer and football are coming out with more similar automobile and car are coming out with excuse me are being more similar by just looking at these 2 dimensions and that is some very interesting aspect that the distributional semantic models try to capture.

Now, suppose I go back to my vector representation, so, where I have all the 7 dimensions and I want to find out which 2 words are similar and to what degree. So, one simple thing I can do is to take the dot products. So, that will tell me if they are having weights in similar dimensions if they having weights in different dimensions the dot products will be 0 or close to 0, but if they are having weights in same dimensions their dot product will be high suppose we are not doing any normalization, I can just take simple dot products and find out which 2 objects or which 2 words are more similar than other words. So, if you try to do that in this example, so what did you would you find? Automobile and car, so dot product would be 1 plus 2 3 plus 1 4 and everything else is 0.

Similarly, automobile soccer you will find 0, automobile football 1, car soccer 1, car football 2 and soccer football 5. So, from these simple dot products computations, so what is something that you see? So, you see that immediately you can capture the idea that here automobile and car are very similar. They are having a very high dot product 4, similarly soccer and football are also very similar, they are having a dot product of 5, on the other hand, automobile soccer are not very not much similar car and soccer are not much similar and so on. So, you can find out some sort of differential aspect of my meaning that is which 2 words have similar meanings and which 2 words are different meanings just by looking at the simple distribution and as you keep on getting more and more data you will you will be able to capture more and more information about how similar and how different 2 meanings of words are this is a very very basic idea that that I had tried to give in this lecture.

Now we will see, what are the different formal ways in which we can construct these models, how do we count what are different ways in which we can denote the entries of this matrix and how do we use that to compute similarity across 2 different words. So, this we will cover in more detail in the next lecture, but I hope that the main idea of using these distribution models is clear.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 33**  
**Distributional Models of Semantics**

Welcome back for the second lecture of this week. So, in the last lecture, we started our discussions on distribution semantics and we took a very simple example in the very naive method we constructed that distribution matrix and try to compute the similarity across words. Now you will see this concept much more formally that how do we construct such models and what are the different applications that we can use them all, we will see some examples.

Firstly, let me just go back a step and see why do we need to talk about the distributional semantics? So, if I talk about a vector space model without distributional semantics, what would happen? So, there words would be treated as various atomic symbols. So, what do I mean by that?

Suppose think of your semantic space as corresponding to various different words that you see in your vocabulary.

(Refer Slide Time: 01:19)

A handwritten note on a light blue background. At the top, there are two vectors labeled  $v_1$  and  $v_2$ , each with dimensions  $\dim_1, \dim_2$  and  $\dim_V$ . Below them is a formula for a word vector  $w_i = [0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0]_V$ . A vertical arrow points from this formula to a row vector  $w_i = [0 \ 1 \ 0 \ 0 \ 0]$ . To the left of this arrow, it says  $w_i - w_j = 0$  and  $w_i - w_k = 0$ . To the right, it says "One-hot encoding" with a checkmark. Below the row vector, there is a bracketed list:  $\text{bank} = [2 \ 0 \ 1]^T$ , where  $2$  is under "government" and  $1$  is under "regulation".

If your vocabulary is of size  $V$ , so, contains word with index 1, 2 up to  $V$  and I want to give a representation to these words in the vocabulary some representation. So, what is the representation that I can give without looking at any distributions? So, I will say that each one is a different dimension; dimension 1, dimension 2, up to dimension  $V$  and how do I denote a word in these dimensions?

Suppose I have the  $i$  th word. So, I will say, so let me denote it the  $i$  th word, this will be a vector in  $V$  dimensions where only the  $i$  th element is 1, everything else is 0. So, like that I can give a different representation to each of the  $V$  words where only 1 element corresponding to the index will be 1, everything else will be 0 and this is also called the one-hot encoding, only one of the entry is 1, everything else is 0.

Now, so there is also my semantic space where all the  $V$  dimensions correspond to different  $V$  different words. Now what is the difference here? I am not capturing distribution, I am just saying this word is the  $i$  th dimension nothing else. Now can I do semantics with that or can I capture the meaning of the words with that? Each word has a different vector, now suppose I want to find out if  $w_i$  and  $w_j$  are more similar than  $w_i$  and  $w_k$ . So, what is the similarity between  $w_i w_j$  and  $w_i w_k$ ?

What will happen if I use this encoding? So, I will find out because here the  $i$  th element will be 1, everything else will be 0, here  $j$  th element will be 1, everything else will be 0. If I try to take a dot product, I will get a 0 in both the cases, even if you find that these 2 words are looking similar than these 2 words and that is 1 problem with using on the one hot encoding.

(Refer Slide Time: 03:39)

The slide has a blue header bar with the title "Vector Space Model without distributional similarity". Below the header, the text "Words are treated as atomic symbols" is displayed. A light blue box contains the text "One-hot representation". Below this, two vectors are shown: "motel [0 0 0 0 0 0 0 0 0 1 0 0 0 0]" and "hotel [0 0 0 0 0 0 0 1 0 0 0 0 0 0]". The text "AND" is between the vectors, and "= 0" is at the end. At the bottom of the slide, there is a navigation bar with icons and the text "Ptwan Goyal (IIT Kharagpur)", "Distributional Models of Semantics", "Week 7, Lecture 2", and "2 / 19".

Suppose I take 2 words here like motel and hotel and suppose motel occur at this index, it will have 1 everything as 0, hotel occurs this syntax, this is 1 everything as 0. So, if I take to try to take the dot product of these 2, I will get a 0 and if I take hotel and book that also will give me a dot product of 0. So, this will not capture that hotels and motels are much more similar than hotels and book.

(Refer Slide Time: 04:10)

The slide has a blue header bar with the title "Distributional Similarity Based Representations". Below the header, a light blue box contains the text "You know a word by the company it keeps". An orange box contains the text "government debt problems turning into banking crises as has happened in saying that Europe needs unified banking regulation to replace the hodgepodge". A pink box contains the text "These words will represent banking". At the bottom of the slide, there is a video player showing a person speaking, and a navigation bar with icons and the text "Ptwan Goyal (IIT Kharagpur)", "Distributional Models of Semantics", "Week 7, Lecture 2", and "19".

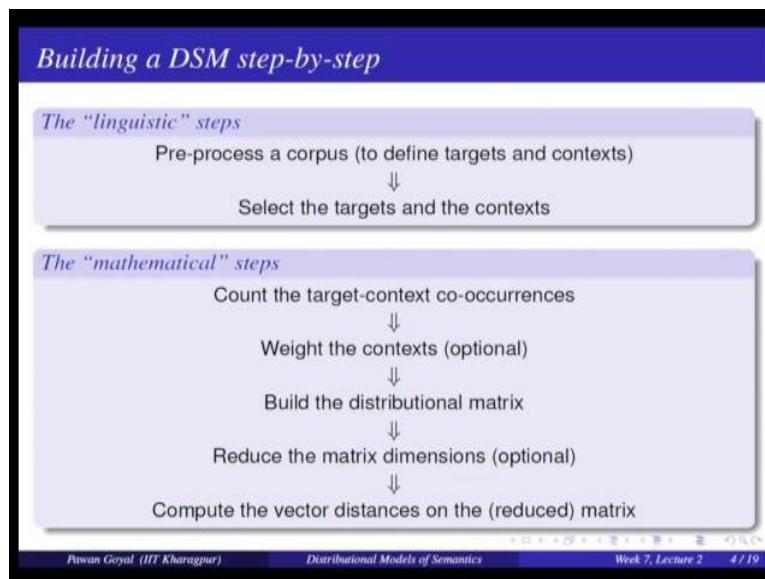
Now what do we do in distribution semantics? We use this idea that you know a word by the company it keeps. So, instead of putting only 1 and everything else 0, try to model the

distributions. So, suppose I take a word like banking, I will not denote it by a simply one at its syntax, but I will see what are the other words that come its context.

If we talk it is occurring in the corpus with government, dept, problems, turning crises, Europe, regulations, replace, etcetera, then these words will be used to represent banking. So, now, what will happen? So, instead of my one hot encoding, I will now go to a distribution representation where I will say for the word like banking, how many times suppose this is my dimension corresponding to government.

How many times it occurs with government? Say 2 times, similarly here it is my dimension corresponding to say regulation, how many times it is occurring with regulation? So, this is I am doing for bank, now I can do similar thing for something else like economy and so on, what they will find? Probably the word banking economies have quite similar distributions than the word like banking and something like play, yes. So, instead of having one-hot encoding I am having a distributed representation and that is what is being done by these distribution semantic models.

(Refer Slide Time: 05:58)



Now, how would I build a distribution semantic model step by step? So, you will take a corpus that is the that is something that you have require that contains a lot of data the various usage of different words in the sentences now you will do some pre processes to define, what are your target words and what are your context words? Context can be context word or

something else, we will see some examples. Now once you select what are your targets and context from your data and what is the next task.

Next you will count, how many times the target co occurs with various context and you will give some weights to these contexts it can be the simply number of times they co occur or some function applied over this number and we will see again see some examples for that. Now once you have given the weights, you build the matrix; distribution matrix now some optional step is if you want to reduce the dimensions of this matrix because the dimensions in general can be very very high, if you are talking about a large corpus, you might have millions of words as your context. So, do you want to further reduce the dimensions, then once you have the representation in that reduce dimension you can also try to capture the similarity across words. So, you compute the vector distances on the reduced matrix and this is the overall pipeline for constructing a distribution semantic model.

(Refer Slide Time: 07:28)

The slide has a title 'Many design choices' at the top. Below it is a grid of four columns:

Matrix type	Weighting	Dimensionality reduction	Vector comparison
word × document	probabilities	LSA	Euclidean
word × word	length normalization	PLSA	Cosine
word × search proximity	TF-IDF	LEA	X Dice
adj × modified noun	PMI	PCA	Jaccard
word × dependency rel.	Positive PMI	IB	KL
verb × arguments	PPMI with discounting	DCA	KL with skew

Below the grid is a section titled 'General Questions' with two bullet points:

- How do the rows (words, ...) relate to each other?
- How do the columns (contexts, documents, ...) relate to each other?

At the bottom of the slide are navigation icons and text: 'Home (Left) (Right) (Up) (Down)' and 'Distributional Models of Semantics Week 2, Lecture 2 3/19'.

Now, here there are many design choices that you might have to make, for example, we said that we will define what are my target words, what are my context words now it can be I there can be many many variations. So, symbol is could be my targets are words and contexts are documents and I am saying how many names this word occurs in this documents and further I can give various weights.

This is 1 matrix type, another could be word cross word how many times this word occurs with another word then again keep on going. So, how many times this word occurs with that

word in certain such proximity or this adjective occurs with this modified noun and word with dependency relation noun and verb with its argument and so on. So, I can have various different sort of matrix types initially we will only focus on the first and second that is my word is the target and document is the context or word is a target and word is the context.

Now, once I have this matrix, what are the other design choices? Then how do I weight the elements? Should I use the probabilities to weight them? Should I use length normalization? TF-IDF, PMI, positive PMI or positive PMI with discounting, what is the method that I should use for weighting the entries in my matrix? Then if I am doing dimensionality reduction, there are the many methods LSA, PLSA, LDA is also one sort of dimension reduction method PCA so on, then once even I have done that how do I compare 2 different vectors?

Once I have built all the vectors in this manner, should I use the Euclidean distance, cosine similarity or dice coefficient, Jaccard coefficient, etcetera. So, they are lot of design choices that you might have to make and each individual on might make a different sort of distribution semantics model, but the underlining idea is the same that you want to capture a distributions from a large corpus.

Now and then there might be different questions that you might be going to answer that once you have built the distribution matrix how do different rows in the matrix, let to each other and how do different columns in the matrix let to each other.

(Refer Slide Time: 09:48)

*The parameter space*

A number of parameters to be fixed

- Which type of context?
- Which weighting scheme?
- Which similarity measure?
- ...

A specific parameter setting determines a particular type of DSM (e.g. LSA, HAL, etc.)

Power Point (MPT Kharragupta) Distributional Models of Semantics Week 2, Lecture 2 8/219

What we have seen a number of parameters that we need to fix like what type of context should I use? What weighting schemes are used? Yes and what similarity measures should I use and different models might be different settings of these parameters now.

(Refer Slide Time: 10:07)

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10
against	0	0	0	1	0	0	3	2	3	0
ago	0	0	0	1	0	3	1	0	4	0
agent	0	0	0	0	0	0	0	0	0	0
ages	0	0	0	0	0	2	0	0	0	0
ago	0	0	0	2	0	0	0	0	3	0
agree	0	1	0	0	0	0	0	0	0	0
ahead	0	0	0	1	0	0	0	0	0	0
ain't	0	0	0	0	0	0	0	0	0	0
air	0	0	0	0	0	0	0	0	0	0
aka	0	0	0	1	0	0	0	0	0	0



Praveen Choudhary (IIT Kharagpur) Downloaded from eGATEWAY

Let us start with the simplest case where words are the targets and documents are the context. Now here is a simple example. So, you are seeing some words as a rows and documents as columns and what do the entries denote that against occurs in the document d 4 1 times in d 7 3 times, d 8 2 times d 9 3 times, as per the document it occurs, it does not occurs and that is how you are represent giving a representation to the word against these doing without any weighting just simply the row counts and this you can be very easily find for any different any word. So, in information table also this is done for each word how many times it occurs in various documents.

(Refer Slide Time: 10:52)

	against	age	agent	ages	ago	agree	ahead	ain't	air	aka	al
against	2003	90	39	20	88	57	33	15	58	22	24
age	90	1402	14	39	71	38	12	4	18	4	39
agent	39	14	507	2	21	5	10	3	9	8	25
ages	20	39	2	290	32	5	4	3	6	1	6
ago	88	71	21	32	1164	37	25	11	34	11	38
agree	57	38	5	5	37	627	12	2	16	19	14
ahead	33	12	10	4	25	12	429	4	12	10	7
ain't	15	4	3	3	11	2	4	166	0	3	3
air	58	18	9	6	34	16	12	0	746	5	11
aka	22	4	8	1	11	19	10	3	5	261	9
al	24	39	25	6	38	14	7	3	11	9	861



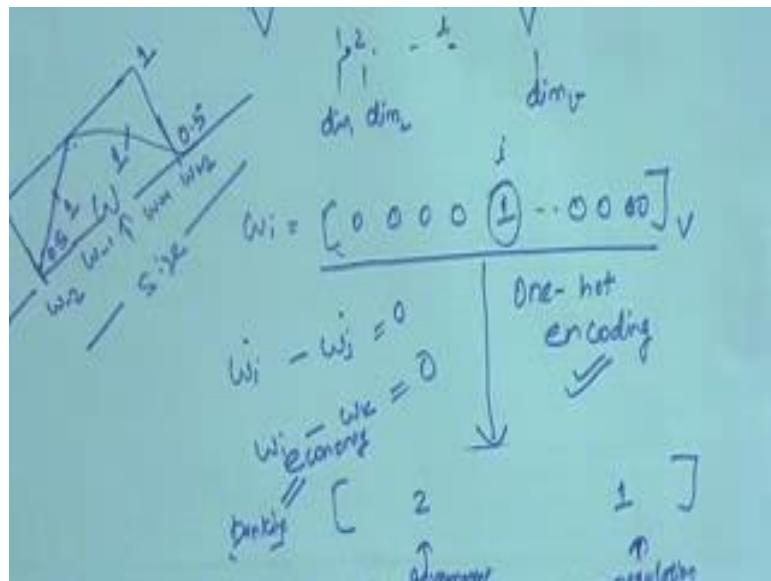
Photo Credit: MIT Bangalore  
Circumlocution Models of Semantics  
Week 2, Lec 2

On the other hand suppose your contexts are the word then you are trying to show how many times a word occurs with another word. So, in this particular case, so the word against occurs with age 90 times, age agent 39 times and so on and this diagnoses tells how many times the word agent occurs. So, it is being computed by as if co occurring with itself. So here so, again you will define a context window, it can be a sentence or something and you will find out, how many times the word air occurs with against and age occurs with age and so on, this you can do for all the words or a specific set of word that you have already chosen for your analysis.

This is simply the count, now if I am using my words as contexts; document contexts are easy because I know, what is the context window size? This is the whole document. So, I will say whether these words occur in the document or not or how many times it occurs? Now suppose I am using words at constant now, now the question comes in what should be the size of my context window should I choose only the co occurrence within 2 words occurring or within a paragraph, within the whole document and this is the design choice that you can make, depending on are you trying to measure similarity on based on some very close co occurrences or you are allowing even a some very very distant co occurrences in the same document. So, what is the parameter here? What is a size of my window? So, how far am I looking it and what is the shape of my window? Is it rectangular, triangular or something else? Let me just briefly let what do I mean by the shape of the window.

So, that is suppose I am having a word w.

(Refer Slide Time: 12:50)



And it has some context. So, I am trying to see how many times this word occurs with these words around the context. So, suppose this is the previous word, previous to previous word, next word, next to next word, so my size is am I looking at only 2 words around it or more than that 3, 4 and so on, this is my size. So, this is the size and what do I mean by shape? So, we said it can be triangular. So, triangular will be something like this is rectangular, what is the difference in triangular?

What I will do as I keep on moving away from my target word, I reduce the count, I will say this count, I will treat as 1, this I will treat as 0.5. So, this is the strength is decrease as you go away from the target word similarly this can be one this can be point 5 and you can as such were give it any shape like exponential or whatever, in that way you can capture even much further co occurrences in that (Refer time: 14:13) you know what you will what you will do you will say I will count each co occurrence size same way. So, each has a weight of 1 or whatever you give.

This is the idea of window size and window shape.

(Refer Slide Time: 14:27)

The screenshot shows a software application window titled "Words as contexts". Under the heading "Parameters", there are two bullet points: "Window size" and "Window shape - rectangular/triangular/other". Below this, a section titled "Consider the following passage" contains a news clipping. The clipping discusses suspected communist rebels killing in Makati, Philippines, mentioning the New People's Army, a commandeered passenger jeep, and soldiers/police killed since January. The words "suspected", "communist", "rebels", "killed", "July", and "1989" are highlighted in blue, indicating they are part of the window centered on the word "rebels".

Now let us take 1 example from some news article. So, this is my passage the suspected communist rebels on 4th July 1989 killed and so on, this is my passage and I want to capture the co occurrence over there are. So, what I will do? I will first define, what is my window size? Suppose I have a window of size 5, I want to use window of size 5, where I am taking 2 words either side of my target word.

What will the window look like? Take the word like rebels here I will find out its co occurrence only with 2 words on the left and 2 words on the right. So, here suspected and communist come here on and 4 come here, similarly when rebels occur here says and communist come here, have and killed come here. So, these are my context words in the window.

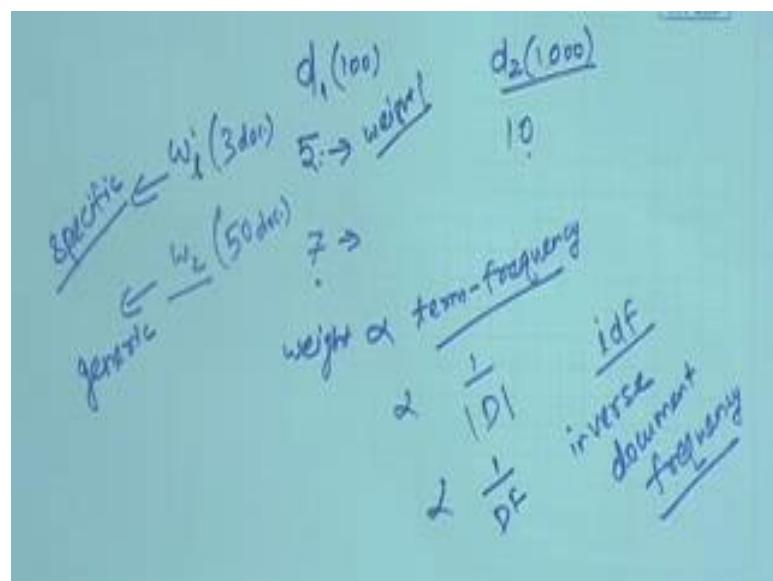
I will only define rebels coming with these words, everything else I will not, I will count as 0. So, this is my unfiltered window, I am not filtering any (Refer Time: 15:43) words, I can also use a filtered window where I can filter the (Refer Time: 15:48) words. So, something like that. So, I will take 2 words either side of the target word, but now these are filtered. So, that they do not contain the stop words.

Here on the left I have the 2 words suspect and communist, but on the right I remove the words on 4 and a number like 1989 and I am taking the words July and killed similarly here I have says and communist in my window, but not have is like a stop word. So, I have removed

that and then killed is there up to 65 is not there and soldiers. So, you can take either the filtered window or an unfiltered window.

Now, how do I weight the context? So, again let us take the 2 cases when my documents are context was is when my words are context. So, when my documents are context how should I weight the occurrence of a word in my document? So, what it should depend on? So, in how many times a word is occurring in the document? What it should depend on or what the weight should depend on?

(Refer Slide Time: 16:59)



I am denoting a word in documents and I am going to give a weight suppose it occurs 5 times in this documents and another word occurs seven times and now I am convert that into some weight. So, what should this weight depend on? So, what can be the different parameters on which it can depend? Firstly, if a word occurs more number of times, it should get a high weight. So, this weight should be proportional to number of times, it occurs and we call it the term frequency how many times a word occurs in a document.

What else if the document is very very long. So, you might have take that window count. So, suppose I have a document  $d_1$  of size 100 and I have a document  $d_2$  of size thousand and I know that word one occurs in  $d_1$  with 5 times and it occurs in  $d_2$  ten times I cannot simply say that in document 2 word, 1 has a high weight because document 2 itself is very very long. So, the weight should be inversely proportion to the length of the document as the length of the document increases the weight should also reduce and what can be the other

measure another measure is very very important also you can even know that if you have heard about some information debate topics.

This is called the inverse document frequency. So, that is IDF inverse document frequency and what is the idea. So, idea is if a word occurs in many many documents through my corpus was such if a words occurs in only selective documents whom should I give a more weight. So, that is suppose my word w one occurs in this document 5 times and w 2 occurs seven times that is, but suppose I also find that w 1 occurs only in 3 documents overall and w 2 occurs in 50 documents.

What can I say about the nature of these 2 words? So, one thing I can say is that w 2 is probably a very generic term and w one is a very specific term this becomes a generic term and this is a specific term. So, idea is that specific term might have more information about the document than a generic term because generic term might be occurring in many many different documents. So, I want to give a high weight to w one because it occurs in very few documents and it has occurred here so; that means, I will give a weight proportional to one divide by number of documents that it occurs in. So, I can call it a document frequency if the document frequency is high I give it a small weight.

(Refer Slide Time: 20:38)

*Context weighting: documents as context*

*Indexing function F: Essential factors*

- **Word frequency ( $f_i$ ):** How many times a word appears in the document?  
 $F \propto f_i$
- **Document length ( $|D_i|$ ):** How many words appear in the document?  
 $F \propto \frac{1}{|D_i|}$
- **Document frequency ( $N_j$ ):** Number of documents in which a word appears.  
 $F \propto \frac{1}{N_j}$

*Indexing Weight: tf-idf*

- $f_i * \log\left(\frac{N_j}{N}\right)$  for each term, normalize the weight in a document with respect to  $L_2$ -norm.

Navigation icons at the bottom include: Home, Back, Forward, Stop, Refresh, and Help. The footer contains: Powerpoint (MIT Kortenbosch), Unsupervised Models of Documents, Week 5, Lecture 2, and 24/19.

These are 3 different parameters or factors on which my function bit will depend on how many times a word appears in the document then what is the different number of words that occur in a document and how many different documents a word occurs in and as we saw, its

weight should be proportion to the number of times that occur in the documents should be inversely proportion to the other 2 factors and there are various different induction functions that have been proposed that take into account all these ideas and one very common measure that is used a TF-IDF that is I give a weight of proportional to the frequency  $F_{ij}$  times log of  $n$  by  $n_j$ . So,  $n_j$  is the document frequency. So, if it is higher, I give it a smaller weight and where  $t$  is the document coming in to picture document length. So, because once I compute the  $t_f$  IDF for each individual term, the document I take the  $L_2$  norm. So, if there are lots of terms, the document, the individual weights will be reduced.

This is 1 very commonly used induction function called TF-IDF I see what are the number of times that word occurs in the document and how many different documents that occurs in. So, 1 thing, they have many variations that have been proposed for this function this is the simplest function that has been known for TF-IDF and finally, we have to take the  $L_2$  norm, this is something that we should not forget that I have to finally, normalize all the different values in a single document. So, that some of the squares adds up to 1. So, now, suppose I take words as my context instead of documents then what is the interesting weighting function I can use?

(Refer Slide Time: 22:33)

*Context weighting: words as context*

*basic intuition*

word1	word2	freq(1,2)	freq(1)	freq(2)
dog	small	856	33,338	490,680
dog	domesticated	29	33,338	918

**Association measures** are used to give more weight to contexts that are more significantly associated with a target word.

- The less frequent the target and context element are, the higher the weight given to their co-occurrence count should be.  
⇒ Co-occurrence with frequent context element *small* is less informative than co-occurrence with rarer *domesticated*.
- different measures - e.g., Mutual information, Log-likelihood ratio

Let us take an example. So, here my target words is word is dog in both the cases context word is small and domesticated 2 different words as my context and what you been shown here. So, target word has the same frequency yes, but the context words in one case is small

as if it sorry small has a frequency of 490580 and domesticated has a frequency of 980. So, you can clearly see here see here that the word small is very very common and domesticated is very specific term.

Now, suppose I find out the co occurrence the dog occurs with small 855 times and dog occurs with domesticated 29 times, now my question is what number I should used denote, how much dog co occurs with the word like small and domesticated. So, if I do not give any weights what will happen here I will say dog occurs a lot with the small 855 times and dog occurs very rarely with domesticated 29 times, but that is not capture the whole picture.

What is the whole picture domesticate is the very very rare word, it occurs only in 900 documents out of which in 30 documents occurs with dog a small occurs in 490000 documents out of which only in 855 documents, it occurs with dog. So, can I use the idea that if a word is rare word, its co occurrence with the target word should have higher weights than if the word is very very common word? So, that is can I use the frequency of the individual words also when I give the weights to their co occurrence and that is what we do why using various association measures that I used to give various weights to the context and the idea is that the less frequent target and context elements are the higher the weight you give to their co occurrence count.

In this case, what will happen? So, because my context element is less frequent here in domesticated their occurrence with dog should be given a high weight. So, co occurrence with the frequent context element small will be less informative here than the co occurrence with the rarer word domesticated and there are various measures that captures that like mutual information is very very popular measure and also log likelihood ratio etcetera they try to capture this. So, how common and rare or rare the words that among which I am finding co occurrence are and how many times they co occurred together. So, they try to use both of these into a single function.

(Refer Slide Time: 25:12)

The slide has a blue header bar with the title "Pointwise Mutual Information (PMI)". Below the header, there is a white background with mathematical formulas. The formulas are:

$$PMI(w_1, w_2) = \log_2 \frac{P_{\text{corpus}}(w_1, w_2)}{P_{\text{ind}}(w_1, w_2)}$$
$$P_{\text{corpus}}(w_1, w_2) = \log_2 \frac{P_{\text{corpus}}(w_1, w_2)}{P_{\text{corpus}}(w_1)P_{\text{corpus}}(w_2)}$$
$$P_{\text{corpus}}(w_1, w_2) = \frac{\text{freq}(w_1, w_2)}{N}$$
$$P_{\text{corpus}}(w) = \frac{\text{freq}(w)}{N}$$

At the bottom of the slide, there is a navigation bar with icons for back, forward, search, and other presentation controls.

What is my, what is the function for point wise mutual information for 2 different words; w 1 w 2? I will find out in mutual information.

What is the probability that they co occurred together in the corpus? I will divide it by what is the probability that they would have co occurred together had they been occurring independently. So, this mutual information tells me, how much information do I gain by seeing? How many times they are co occurring that I would not have obtained by their random co occurrences. So, that is the formula. So, what is their probability of they co occurrence in the corpus divided by what is the probability of a co occurrence in the corpus if they were independent now how do we capture the probability of their occurring of the occurring independently in the corpus.

That would be I will say the probability with which w one occurs and among those times w 2 could have occurred again with this probability. So, it will be the probability of occurrence of w one times probability of occurrence of w 2. So, so this is a function that I can use probability with which w 1 w 2 co occur divide by probability with which w 1 occurs times probability with which w 2 occurs and I will take a log of for that and how do I capture probability of occurrence in a in the corpus for 2 words number of times they co occurred together is a simple co occurrence count divided by n.

N here would denote the different number of context that you can see or in or it can be also the number of different tokens that you have in your data similarly probability corpus of the

word w is very simple this is the unigram model number of times this word occurs divided by all total different number of token that you have seen.

(Refer Slide Time: 27:19)

The slide has a blue header bar with the title 'PMI: Issues and Variations'. Below the header is a light purple box containing the text 'Positive PMI' and 'All PMI values less than zero are replaced with zero.' At the bottom of the slide is a dark blue footer bar with the text 'Powerpoint (MS SharePoint)', 'Unsupervised Models of language', 'Week 7, Lecture 2', and '3.3.1.9'.

In PMI, what would happen in sudden cases? The value might also go to negative values. So, what we what is some variations there? So, you can use up only the positive values and this is called positive PMI, PPMI. So, where only those values that are greater than 0 are taking into consideration everything else is converted to 0.

(Refer Slide Time: 27:46)

The slide has a blue header bar with the title 'PMI: Issues and Variations'. Below the header are two sections: a light purple box for 'Positive PMI' and a pink box for 'Bias towards infrequent events'. The 'Positive PMI' box contains the same text as the previous slide. The 'Bias towards infrequent events' box contains the text 'Consider  $w_j$  having the maximum association with  $w_i$ ,  $P_{corpus}(w_i) \approx P_{corpus}(w_j) \approx P_{corpus}(w_i, w_j)$ '. At the bottom of the slide is a dark blue footer bar with the text 'Powerpoint (MS SharePoint)', 'Unsupervised Models of language', 'Week 7, Lecture 2', and '3.3.1.9'.

Now, there is one problem in this mutual information approach that there is a biased word some in frequent events. So, remember why we came this idea of mutual information we wanted to give high weight to the events that are in frequent if 2 words are rare, we do wanted to give their association a high weights, but what happens if 2 events 2 words are very very rare they get they might get some high weight. So, some one particular cases suppose think about the scenario with  $w_i$  and  $w_j$  have the same occurrence in the corpus as  $w_i w_j$  together now what to do the probability or the mutual information in this case.

(Refer Slide Time: 28:32)

The image shows a handwritten derivation of the Mutual Information (MI) formula. It starts with the formula for PMI:

$$PMI(w_i, w_j) = \log \frac{P(w_i, w_j)}{P(w_i) P(w_j)}$$

Below this, it shows two cases for the joint probability  $P(w_i, w_j)$ :

- If  $w_i = w_j = f_{ij} = 1$ , then  $P(w_i, w_j) = \frac{1}{N+1} \cdot \frac{1}{N+1} = \frac{1}{N^2+2N+1}$ .
- If  $w_i = w_j = f_{ij} = 10$ , then  $P(w_i, w_j) = \frac{10}{N+1} \cdot \frac{10}{N+1} = \frac{100}{N^2+2N+1}$ .

It then notes that  $w_j$  occurs once also with  $w_i$ , and  $P(w_i) = \frac{1}{N}$ . The final result is given as  $P(w_i, w_j) = \frac{1}{N}$ .

So, PMI between  $w_i w_j$  is log of probability  $w_i w_j$  divided by probability  $w_i$  times probability  $w_j$  and suppose all 3 are equal. So, what would happen? This would cancel. So, this will be log 1 by  $p_{w_i}$ . So, what it is saying if these 3 are equal PMI will be high, if PMI is low and immediately that that gives a bias in the favor of events that are smaller though or words that occur very very rarely. So, you can think of 2 scenarios where all this 3 probabilities are 1 by 100 or versus all these 3 probabilities are 1 by 10000. So, either we I would like the association to be the same in both the cases, but what would happen in this formula the association become high when the probability is low and this is not desired. So, this will create a problem when their individual probabilities are very very low, this immediately will become PMI will become very very high.

And there are another case where you can see that if the word  $w_j$  occurs only once in the corpus and it also occurs that time in with  $w_i$ . So, what would happen? So,  $P_{w_i w_j}$  and  $P_{w_i}$

j will become similar here also because what I am saying w j occurs once also with w i. So, P w j will be 1 by n and P w i w j will also be 1 by n. So, if I put that here it will again come out to be this formula and; that means, it will the PMI will depend on w i if w i is frequent PMI will be low if it is rare it will become high I mean this is not desire.

To evaluate this problem, so they are 2 different things that you can do, one is you can probably start by the moving all the words that are having very very low occurrences. So, all the words that are occurring less than 2 or 3 times you can remove from your data all together and the words that are occurring more than that this problem will be not that bad, but if you want took also take into considerations the very very infrequent words. So, what you might have to do? You might have to takes into consideration some sort of bias.

(Refer Slide Time: 31:16)

What is the idea? There are these very important discounting factors that are proposed by Pantel and Lin. So, what it says that you multiply the PMI value with this discounting factor that is  $f_{ij}$  divided by the  $f_{ij}$  plus one times minimum of  $f_i$   $f_j$  divided by minimum of  $f_i$   $f_j$  plus 1. So, what would happen now? So, you will multiply this with PMI. So, if a frequency of the individual events are very very small this factor will become smaller. So, like that is take 2 cases here, one where  $f_i$  is equal to  $f_j$  is equal to  $f_{ij}$  is 1, where they are even second  $f_i$  is equal to  $f_j$  is equal to  $f_{ij}$  is equal to 10 not. So, rare, so in the first case what will be the discounting factors 1 divided by 1 plus 1 times 1 divided by 1 plus 1 this will be 1 by 4 in this case it will be 1 divided by 10 plus 1 times 1 divided by 10 plus 1 this will be sorry 10

divided by that. So, this will 100 divided by 121. So, discounting will be much more higher here. So, you will divide the PMI by 0.25, here you will or multiply by 0.25, here you multiply by roughly 0.8 and this takes care of the problems that we had talked about in with infrequent events.

Whenever there are infrequent PMI will become very high. So, if you use this discounting it will again come back to a reasonable value and this will not create a problem when the events are not so rare.

(Refer Slide Time: 32:35)

Distributional Vectors: Example	
Normalized Distributional Vectors using Pointwise Mutual Information	
<b>petroleum</b>	oil 0.032 gas 0.029 crude 0.029 barrels 0.028 exploration 0.027 barrel 0.026 spec 0.026 refining 0.026 gasoline 0.026 fuel 0.025 natural 0.025 exporting 0.025
<b>drug</b>	trafficking 0.029 cocaine 0.028 narcotics 0.027 hda 0.026 police 0.026 abuse 0.026 marijuana 0.025 crime 0.025 columbus 0.025 arrested 0.025 addicts 0.024
<b>insurance</b>	insurers 0.028 premiums 0.028 lloyds 0.026 reinsurance 0.026 underwriting 0.025 pension 0.025 mortgage 0.025 credit 0.025 investors 0.024 claims 0.024 benefits 0.024
<b>forest</b>	timber 0.028 trees 0.027 land 0.027 forestry 0.026 environmental 0.026 species 0.026 wildlife 0.026 habitat 0.026 tree 0.025 mountain 0.026 river 0.025 lake 0.025
<b>robotics</b>	robots 0.032 automation 0.029 technology 0.028 engineering 0.026 systems 0.026 sensors 0.025 welding 0.025 computer 0.025 manufacturing 0.025 automated 0.025

Here is 1 example of what kind of vector do you get by taking this PMI. So, on the left hand side you have, so, what is done here? You taking a large corpus and finding out for individual words what are the other vectors that I having high PMI values and then you are doing some normalization. So, that all the PMI value is on this course for a given word add up to 1. So, what do you see here?

If I take a word like petroleum, you find words like oil, gas, crude, barrels, exploration that are coming out to be having very high PMI values, with drug you find words like trafficking, cocaine, narcotics, insurance, you find a words like insurers, premiums, Lloyds, with forest you find the words like timber, trees, land and robots; robotics you find word like robots, automation and so on and all this is coming out just by putting this PMI function over this corpus where different words and sentences are put together.

I am computing this function and finding out for a word what are similar words and you can see that it is capturing very nicely what are the other very similar words to that to s starting word like robotics you find words like robots and automation, but the word like forest you find immediately words like trees etcetera and this can give you nice intuition that you can use that very nicely to capture which 2 words are similar which 2 words are very very different and in some of application you can make use of that. So, I will start from here in the. So, for the next lecture and we will see how we can use that for some very interesting application like term (Refer Time: 34:38) information retrieval, I will define the problem and see how do you use that.

Thank you.

**Natural Language Processing**  
Prof. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 34**  
**Distributional Semantics - Applications, Structured Models**

Hello everyone, welcome to the 3rd lecture of this week. So, we are talking about distribution semantics and we had already discussed what is a generic framework of distribution semantics, how do you build models by taking different sorts of targets and context and how do you fill up all the antigen of matrix. Now in this lecture, we will talk about some of the some applications for distribution semantic models, also how do I compare similarity across two words and then we will move towards some a structured model of distribution semantics.

(Refer Slide Time: 00:56)

*Application to Query Expansion: Addressing Term Mismatch*

*Term Mismatch Problem in Information Retrieval*

- Stems from the word independence assumption during document indexing.
- User query: *insurance cover which pays for long term care*.
- A relevant document may contain terms different from the actual user query.
- Some relevant words concerning this query: *{medicare, premiums, insurers}*

*Using DSMs for Query Expansion*

Given a user query, reformulate it using related terms to enhance the retrieval performance.

- The distributional vectors for the query terms are computed.
- Expanded query is obtained by a linear combination or a functional combination of these vectors.

Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structured Models      Week 7, Lecture 3      2 / 15

Starting with some application, so let us talk about this problem of term mismatch that we see in the case of information retrieval so, what is the problem? So information retrieval, the user is giving a query that is in terms of certain words that he feels are describing his information in it and so the system is trying to match this set of keywords to all the documents that are there in the depository and by doing this matching they are trying to find out what are the potential document that can be returned to the user.

Now, what is one problem here? So, it might happen that the particular concept that the user has in mind; the user expressing by using certain words, but the documents have the same

concept using a very very different word. So, they are, the words are similar on the semantic label, but on the surface label they are 2 different keywords or 2 different words. So, this is called a term mismatch problem and one of the key issues in information retrieval is how do we solve this term mismatch problem? How do I find out that the user typed this particular term? But he might also be interested in the other term that is semantically similar, but on the surface form it is different.

Let us see an example. So, suppose there is a user query insurance cover which pays for long term care and it might happen that the relevant document that contains the answer to the user query may have words like Medicare, Premiums, Insurers, etcetera that are not directly provided in the user query. Now, one question is; how do I find out that these documents that contain some similar words are also relevant to this user query.

So, for this task, one can use issue semantic models for query expansion. So, query expansion is one of the techniques that is used for this term mismatch problem. So, idea is that if the user has given a query with certain words can I try to find out some other words that are semantically similar to the user query words? If I find out some words then I append those words to the user query and this is called the expended query expansion process and now I give this expended query to my search engine and then retrieve the documents.

Now, what we will see in brief, can we use our distribution semantic models for this problem of query expansion. So what is the idea? So, once user gives a query, we reformulate it using the terms that are relevant or that are related to the terms that are already there in the query, find out some related terms and try to use that to improve the retrieval performance. So, how do we find out the related terms? So firstly, I have the query terms that the user is giving, find out what are the distributional vectors and take a function combination of all the distributional vectors and obtain the expanded query.

Suppose that the query has 3 terms find out what are the related terms to all this 3 terms in the distributional sense and make a linear combination of all such possibilities and give the expanded query to the user.

(Refer Slide Time: 04:14)

### Query Expansion using Unstructured DSMs

**TREC Topic 104: catastrophic health insurance**

**Query Representation:** surtax:1.0 hcfa:0.97 medicare:0.93 hmos:0.83 medicaid:0.8 hmo:0.78 beneficiaries:0.75 ambulatory:0.72 premiums:0.72 hospitalization:0.71 hhs:0.7 reimbursable:0.7 deductible:0.69

- Broad expansion terms: **medicare, beneficiaries, premiums ...**
- Specific domain terms: **HCFA** (Health Care Financing Administration), **HMO** (Health Maintenance Organization), **HHS** (Health and Human Services)

**TREC Topic 355: ocean remote sensing**

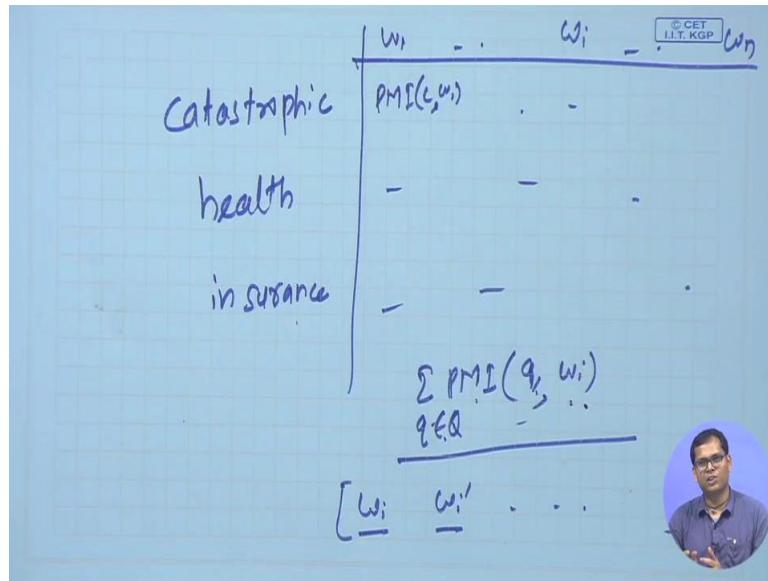
**Query Representation:** radiometer:1.0 landsat:0.97 ionosphere:0.94 cnes:0.84 altimeter:0.83 nasda:0.81 meterology:0.81 cartography:0.78 geostationary:0.78 doppler:0.78 oceanographic:0.76

- Broad expansion terms: **radiometer, landsat, ionosphere ...**
- Specific domain terms: **CNES** (Centre National d'Études Spatiales) and **NASDA** (National Space Development Agency of Japan)

Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structures      Week 7, Lecture 3      3 / 15

How will it look like? So, suppose the user gives a query like catastrophic health insurance. So, what you are doing? So, in your distribution vector, suppose you are finding out, what are the terms that are related to each of the 3 terms catastrophic health and insurance?

(Refer Slide Time: 04:34)



What will happen? So, user has given these 3 terms and suppose, you are using words at context. So, the different words were there  $w_1, w_i, w_n$  and for each word. So, you are computing, what is the co occurrence? So, it can be say PMI value, what is the PMI between catastrophic and  $w_1$ . So, like that, you are computing for all the words. So, now, what you

might do? You might say, what are the words that are associated with all the 3 words? So that are having high PMI values. So, one way is you just add all the PMI values PMI  $q \ w \ i$  for all words  $q$  in my query.

For each word  $w$ , why I complete a score like that what is the PMI of that query, one of the query word with this word  $w \ i$  and add your all the query words and then I can sort it and maybe I can normalize it with respect to the highest term. So, what will happen at the end of that if I can sort these course and find out what are the words  $w \ i$ ,  $w \ i$  prime etcetera that are very much related to the query terms and all these we have already covered how do we computed this course, once we have this course I can compute this symbol formula and find out what are the terms in a decreasing order.

This one part way, but you can have any other sort of functional combination. So, this is a simple addition, but you can have multiplication and other sort of function combinations also. So, suppose we do that. So, what happens to the actual query? So, this is my ac actual query catastrophic health insurance and if I try to use this technique to find out some other related terms I get terms like this. So, I have terms like surtax, h c f a, Medicare and hmos, Medicaid, hmo, beneficiaries, premiums and so on. Now so these terms are taken by from a news corpus that is US news corpus. So, they are certain terms that are relevant to the US medical system. So, what are the; so what you are seeing in the expected terms? So some terms are like broad explanation terms so; that means, if you look at may be some (Refer Time: 07:03) or some other thesaurus you can find out such kind of similar terms.

You will say Medicare, beneficiaries, premiums, are some broad explanation terms. So, they are ready to the query. In addition, you also get some very very specific domain term like here HCFA that is US domain term for health care financing administration, similarly HMO for health maintenance organization, HHS. So, these are very very domain specific terms. So, if you know that you query is coming from a certain domain, this approach can be very very helpful to also find out terms that can be used to explain the query and not also a domain specific.

Let us take another example. So, these are so, I am showing here some TREC topics. So, TREC is a text t o conference. So, they organize various competitions for information retrieval. So, they are actual queries from the TREC data. So, you have a query like ocean remote sensing and when you use this method, you can find some other terms like

radiometer, landsat, ionosphere, cnes, altimeter, nasda, meteorology and so on and again as we seen saw in the earlier query, there are some broad explanation terms like radiometer, landsat and ionosphere that are connected to the query and they are some domain specific terms like cnes and nasda here.

This is one very nice application of distribution semantic models you have some existing query you want to match it to some document, why do not you expand this query by using some related terms and this is the idea. And this we can use in general for any rid matching task you are having do different text data and you are trying to match these. So, try to find out if there are somehow related on the semantic label using distribution semantic models and use this idea to find out if they are similar or not.

(Refer Slide Time: 09:04)

## Similarity Measures for Binary Vectors

Now, so once we have computed the distributional semantic model so, I have the vectors for different words. So, different target words, I have the vectors. Now, how do I compute the similarity between different words? So, it depends on what is your representation that you are using for computing the semantics or the what is the vector representation if it is the binary vector you will use different sort of similarity methods than if it is a real valued vectors or if it is a probability distribution you will use a different sort of similarity matrix.

So, as is the framework allows you to use any of these vector representations. So, let us see if you use, if you are using any of these what kind of similarity methods you can use to compute similarity between 2 words. So, let us say my. So, have words X and Y and they are

denoted using some binary vectors. So, as such any sort of distribution you can also convert into binary vectors.

How do I compare, how do I find similarity between 2 word 2 words where the representation is binary vectors? So, for binary vectors, we have some standard methods like using dice coefficient. So, what is that? 2 times intersection of X and Y divided by length of X plus length of Y and what do I mean by this. So, X and Y are binary vectors.

(Refer Slide Time: 10:32)

The handwritten notes show the calculation of the Dice coefficient for two binary vectors, X and Y. Vector X is represented as [1 1 1 ... 1 0 0] (length 19) and vector Y as [0 0 0 ... 0 1 1 ... 1] (length 10). The Dice coefficient formula is given as  $\text{Dice} = \frac{2|X \cap Y|}{|X| + |Y|}$ . Below the formula, the intersection count is calculated as 1 (from the first dimension), and the lengths of X and Y are shown as 19 and 10 respectively. The final result is 0.1. A note at the bottom right says "Convert these to binary".

Suppose X and Y are binary vectors and let us say the size of the my vector; the dimension are say since it is a 19 dimensions and my X is 1 1 1 1 in the first 10 dimension and 0 0 in the rest 9 dimensions. On the other hand Y is 0 in the first 9 dimension and 1 in the last 10 dimensions.

Now, I have 2 vectors - X and Y and I want to compute the similarity between these 2 vectors, suppose I am using dice coefficient. So, the formula is 2 times X intersection Y divided by length X plus length Y. So, this is the length of X intersection Y, now what is X intersection Y? So, that is only 1 element. So, X intersection Y has 1 element that is 1. So, this will be simply 1, 2 times 1 length of X, how many 1s are there.

We will not take the length of the vector because otherwise it will be the same for everything, everything will have the same size. So, length of X means how many entries are 1. So, here it will be 10 for Y, also it is 10 10 plus 10. So, this will be 0.1. So, now, this is if you have the

binary values, suppose you do not have the binary values so, suppose my values are like my X could be 0.3, 0.5, 0.7, 0.01 and so on. So, if you want to use these measures. So, 1 simple way could be convert them to binary and now converting would mean you would put a threshold that if the value is below this threshold then you put it to 0 above this threshold you will put it to 1. So, suppose here threshold is 0.05. So, what you will do? This will go to 0, this will go to 1, this will go to 1, this will go to 1 and here you will check whether less than 0.05 then go to 0 if greater than then go to 1 like that. So, you can convert any such vector into binary representation and then use this dice coefficient, now are there some other methods of computing similarity also?

So, we have Jaccard coefficient and overlap coefficient. So, Jaccard coefficient is X intersection Y divided by X union Y. Now what is the difference between Jaccard and dice, in what scenario Jaccard will give a different value than dice coefficient. So, let us try out the same example. So, I am using the Jaccard as X intersection Y divided by X union y. So, in this case what is X intersection Y this will remain as one only one entry is one and what is X union Y now X union Y will contain all the elements this will be 19. So, what you are seeing for the same 2 vectors dice give the value of point one and Jaccard gives 1 by 19 that is closely 0.05. So, these give much smaller value.

Why is that? So, you are seeing here if there are very small number of shared entries Jaccard further penalizes. So, that is what you are seeing here there are very only 1 entry common among 20. So, Jaccard is giving further penalty.

Now, what will overlap do? So, overlap is X intersection Y divided by minimum of X Y. So, can you think of a scenario where overlap it can becomes 1, but say Jaccard will not become 1. So, this will happen when 1 of X and Y is completely included in the other.

(Refer Slide Time: 14:54)

© CET  
I.I.T. KGP

$$X = [11111 \ 00000]$$

$$Y = [11111 \ 11100]$$

$$\text{Jac.} = \frac{|X \cap Y|}{|X \cup Y|} = \frac{5}{8}$$

$$\text{Overlap} = \frac{|X \cap Y|}{\min(|X|, |Y|)} = \frac{5}{5} = 1$$

Let us say, let us take a different case suppose my  $X$  is 1 1 1 1 1 0 0 0 0 0 and  $Y$  is 1 1 1 1 1 1 1 0 0, so in this case, what would be the Jaccard? It will be  $X$  intersection  $Y$  divided by  $X$  union  $Y$ . So, this will be 5 divided by 8 and what will be overlap? This will be  $X$  intersection  $Y$  divided by minimum of  $X$  and  $Y$ . So, this is again 5 and what you mean of  $X$  and  $Y$  again 5. This will become 1. So, that is if 1 of the vector is completely subsumed by another vector overlap is 1 and that will not happen in Jaccard dice coefficient and similarly if there are a small number of shared entries, Jaccard will give us smaller value. That means, depending on what kind of similarity want to use, you can have either dice Jaccard overlap, suppose in your task, you want to find out if 1 of the words is completely subsumed by another, you will use overlap and not Jaccard, but if you want to see some other criteria you can choose 1 of the 3 methods.

Here so what you have seen? Jaccard coefficient penalizes small number of shared entries while overlap coefficient uses the concept of inclusion where the one of the entries completely included in the other one, now this is a (Refer Time:16:29) vectors of binary vectors.

(Refer Slide Time: 16:31)

Let  $\vec{X}$  and  $\vec{Y}$  denote the distributional vectors for words  $X$  and  $Y$ .  
 $\vec{X} = [x_1, x_2, \dots, x_n]$ ,  $\vec{Y} = [y_1, y_2, \dots, y_n]$

*Similarity Measures*

Cosine similarity :  $\cos(\vec{X}, \vec{Y}) = \frac{\vec{X} \cdot \vec{Y}}{|\vec{X}| |\vec{Y}|}$

Euclidean distance :  $|\vec{X} - \vec{Y}| = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$

Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structures      Week 7, Lecture 3      5 / 15

Suppose they are real number values, so like  $X$  and  $Y$  so, say the  $n$  number of real number values. So, then you can use simple cosine similarity or Euclidean distance.

$$\cos(\underline{X}, \underline{Y}) = \underline{X} \cdot \underline{Y} / |\underline{X} \cdot \underline{Y}|$$

2

$$|\underline{X} - \underline{Y}| = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

You can use cosine similarity and Euclidean distance, now what is the difference between the two? If my vectors are not normalized cosine similarity Euclidean distance are different, but if they are normalized, they will be the same and they will give the same sort of ranking and that you can do a very simple exercise, if they are normalized, they will give me the same sort of ranking between the similarity of vectors.

(Refer Slide Time: 17:14)

*Similarity Measure for Probability Distributions*

Let  $p$  and  $q$  denote the probability distributions corresponding to two distributional vectors.

*Similarity Measures*

- KL-divergence :  $D(p||q) = \sum_i p_i \log \frac{p_i}{q_i}$
- Information Radius :  $D(p||\frac{p+q}{2}) + D(q||\frac{p+q}{2})$
- $L_1$ -norm :  $\sum_i |p_i - q_i|$

Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structures      Week 7, Lecture 3      6 / 15

Now, on the other hand, suppose my distributions are probability distributions. So, I am denoting different vector size probability distribution, my space of the context vectors.

$$D(p||q) = \sum_i p_i \log p_i / q_i$$

Then how do I compute the similarity? I will use different measures that are used for computing similarity or distance in the case of probability distribution. So, what are the common measures? So, you can use KL. KL divergence is so that is sigma i p i log p i by q i. So, this so KL divergence is asymmetric. So, if you use divergence between p and q and q and p; they will come out, they may come out to be different. So, that is why there is a symmetric divergence also that is information radius. So, p and p plus q by 2 and q and q plus p; p by 2, find the KL divergence between these and add this that is information radius and also you can use is a very simple formula like L 1 norm. So, you have p i q i find out the LL 1 norm.

$$\sum_i |p_i - q_i|$$

What is the difference between p i minus q i sum over all i. So, that is you have different sort of representation and you can use different similarity value similarity matrix.

(Refer Slide Time: 18:24)

**Attributional Similarity**  
The attributional similarity between two words  $a$  and  $b$  depends on the degree of correspondence between the properties of  $a$  and  $b$ .  
Ex: dog and wolf

**Relational Similarity**  
Two pairs  $(a, b)$  and  $(c, d)$  are relationally similar if they have many similar relations.  
Ex: dog: bark and cat: meow

Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structures      Week 7, Lecture 3      7 / 15

Now, let us talk about what are the different; what are some other sorts of distribution semantic models that all that also try to use some specific instructions in the sentences and we try to motivate them using this example. So, that is, what is the difference between an attributional similarity task and a relational similarity task? So, what is attributional similarity? So, that is I am given 2 words like dog and wolf and I want to find out how similar they are. So, similarity between dog and wolf will depend on how much their attributes are similar that is why it is called attributional similarity and by using distributional semantics how do you capture that? What are the other words they are co occurring with? Are they co occurring with similar sort of words? If they are co occurring with similar sort of words, they will have a high attribution similarity and what is relation similarity? So, that is slightly different.

That is now I am talking about pairs. So, that is I have 2 pairs  $a$   $b$  and  $c$   $d$  are they relationally similar that is how many co similar relations that they have. So, example would be like 1 pair is dog and bark second is cat and meow. So, now, dog and bark do they share similar sort of relation as cat and meow. So, this is simple different type of task and so this that is why first one is called attribution similarity. How much the attribution similar among the 2 words? Second is called relation similarity. I have the pair the relation between this pair does that hold also for the other pair and this also gives tries to many analogy testing task. So,  $a$  is to  $b$  as  $c$  is to what? So, we will see how do we extend our distributional semantic similarity or semantics models to also capture all these cases.

(Refer Slide Time: 20:15)

## Relational Similarity: Pair-pattern matrix

**Pair-pattern matrix**

- Row vectors correspond to pairs of words, such as *mason: stone* and *carpenter: wood*
- Column vectors correspond to the patterns in which the pairs occur, e.g. *X cuts Y* and *X works with Y*
- Compute the similarity of rows to find similar pairs

**Extended Distributional Hypothesis; Lin and Pantel**

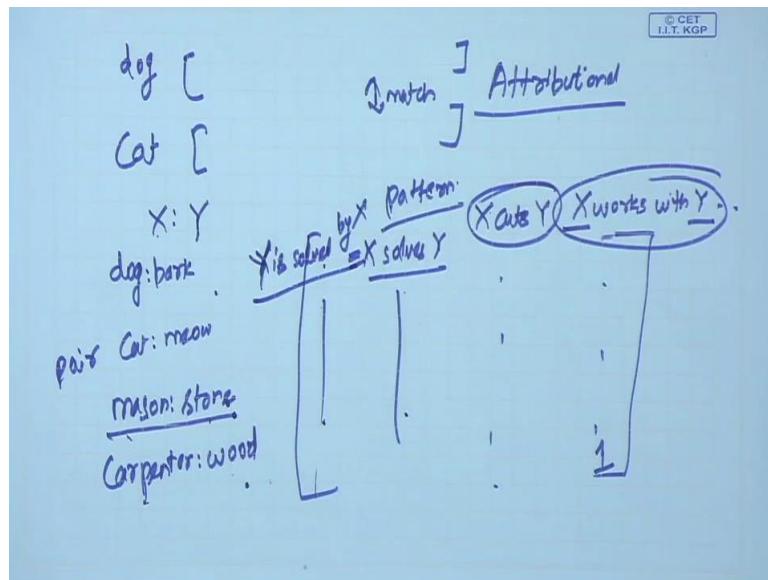
Patterns that co-occur with similar pairs tend to have similar meanings.  
This matrix can also be used to measure the semantic similarity of patterns.  
Given a pattern such as "X solves Y", you can use this matrix to find similar patterns, such as "Y is solved by X", "Y is resolved in X", "X resolves Y".

Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structures      Week 7, Lecture 3      8 / 15

Now, for that we will talk about a different sort of matrix and this will be called pair pattern matrix. So, till now, you are talking about target context matrix. So, let us see if you can use a similar idea for building a pair pattern matrix. So, what do I mean by this?

Here the row vectors will correspond to various pairs of words like *mason stone* and *carpenter wood* and the column would be various patterns in that in the sentences these words occur with. So, like here *X cuts Y*, *X works with Y* etcetera and then you compute the similarity of rows to find similar pairs of words. So, now so, what do I mean by this?

(Refer Slide Time: 21:01)



Till Now, what we were doing we had words like dog cat and I was finding out these representation in various contexts. So, I am finding out vectors for dog vector for cat and trying to match these and this was my simple attributional similarity. So, now, what am I doing I am trying to compute a similarity between pairs. So, now, my rows are different pairs. So, like the pairs can be say dog bark cat meow or it can be like here mason stone carpenter wood. So, like that they can be various pairs now these are my rows now what do the columns denote now columns would be various patterns. So, pattern could be like X cuts Y X works with Y and so on, these are various patterns now what are X and Y you can think of this as my patterns X Y sorry pairs X Y.

Now, what ha how will I fill this matrix simple way would be I have X and Y X can take value like mason and Y can take stone for a given pair now I will go through my coppers and see what are the various patterns in which these words co occur with. So, suppose there is a sentence mason works with or mason works with stone or carpenter works with wood let us take a sentence carpenter works with wood. So, here, I have pattern X works with wood X fits for carpenter Y fits for wood. So, I will say there is a plus 1 here, similarly there will be all sorts of pattern here in which X and Y can occur and I will fill which pair occurs with what patterns. So, now, this is my pair pattern matrix pair pattern matrix and then I can once I have this matrix I can compute which pairs occur in similar patterns and then they are called relationally similar they have similar relation at the other pair.

Then, so we can talk about the extended distributional hypothesis this that much given by Lin and Pantel. So, what is the idea patterns that co occur with similar pairs tend to have similar meanings. So, here we are talking about in terms of our columns. So, here suppose what they are saying if a pattern p one and pattern P 2 if they co occur with similar sort of pairs then they are giving similar sort of meanings.

And therefore, I can use this matrix to compute semantic similarity of patterns also. So, I can find out the semantic similarity of the pairs also the patterns. So, suppose I am given a pattern like X solves Y and I want to find out what are other similar patterns as X solves Y, how will I do that? I will first enumerate all the possible patterns that can occur with X and Y then I find out all the possible pairs how many times they co occur with various patterns. So, I will fill this matrix and then I will find out for this pattern like X solves Y what are some other patterns that us that are having similar pairs as X solves Y.

And what are patterns like X is solved by Y sorry, Y is solved by X, suppose it occurs with similar pairs as X solves Y, I will say that this and this are same and that will be the idea it is like here Y is solved by X Y is resolved in X and X resolves Y. So, what do you find all these patterns occur with similar pairs? So, they can be also called similar.

(Refer Slide Time: 25:17)

**Structured DSMs**

**Basic Issue**

- Words may not be the basic context units anymore
- How to capture and represent syntactic information?  
*X solves Y and Y is solved by X*

**An Ideal Formalism**

- Should mirror semantic relationships as close as possible
- Incorporate word-based information and syntactic analysis
- Should be applicable to different languages

Use Dependency grammar framework

Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structures      Week 7, Lecture 3      9 / 15

Now, what is 1 thing? So, for dealing with this pair pattern matrix words will not be my basic context units. So, how do I capture and represent this sort of information like X solves Y and Y resolved by Y X, how do I capture this information? So, for that I will need a formalism that can capture semantic relations and also various syntactic information can be captured and for that we will go back to our dependency based formalism to capture this kind of information what is the idea.

(Refer Slide Time: 25:59)

## Structured DSMs

Using Dependency Structure: How does it help?  
The teacher eats a red apple.

```
graph LR; eats((eats)) -- "nsubj" --> teacher((teacher)); eats -- "dobj" --> apple((apple)); teacher -- "det" --> The((The)); apple -- "det" --> a((a)); apple -- "amod" --> red((red))
```

- 'eat' is not a legitimate context for 'red'.
- The 'object' relation connecting 'eat' and 'apple' is treated as a different type of co-occurrence from the 'modifier' relation linking 'red' and 'apple'.

Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structures      Week 7, Lecture 3      10 / 15

Let us say I have a sentence like the teacher eats a red apple. So, I can first formulate a dependency graph for this sentence, now I can use this dependency relations to say that only some sort of dependency relations are interesting and others are not interesting. So, for example, I can say that eat is not a legitimate context for red although eats and red co occur with the very small distance context window, I can say that the relation det object and a modifier together do not form a very nice context.

I will not use this co occurrence. So, I can be selective in choosing, what kind of co occurrence information, I will use and what kind of co occurrence information I will not use and I may also give them different sort of weights. So, for example, I can say that the object relation connecting eat an apple will be different than the modified relation connecting red and apple. So, this relation det object and a modifier at can be different relations. So, till now what was happening I was only seeing if this word co occurrence with this word or not.

(Refer Slide Time: 27:36)

*Structured DSMs*

*Structured DSMs: Words as 'legitimate' contexts*

- Co-occurrence statistics are collected using parser-extracted relations.
- To qualify as context of a target item, a word must be linked to it by some (interesting) lexico-syntactic relation

Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structures      Week 7, Lecture 3      11 / 15

Now we have started talking about in different terms this word co occurs with another word in this context. So, with using a det object relation with using an adjective modifier relation and so on, now so from the parser I can get all these relations. So, how do I further use those? So we will say that to qualify as a context a word must be linked by some interesting lexicon syntactic relation. So, what do I mean by lexicon syntactic relation a good dependency path should adjust between the 2 words. So, in simple terms I can only use of use a single edge between 2 words, but in general you can also talk about larger path larger length of edge between the 2 words. So, let us take it simply for the simple paths of length 1 between 2 words.

(Refer Slide Time: 28:11)

## Structured DSMs

### Distributional models, as guided by dependency

Ex: For the sentence 'This virus affects the body's defense system.', the dependency parse is:

The diagram illustrates a dependency parse for the sentence 'This virus affects the body's defense system.'. The words are represented as nodes, and dependencies are shown as directed edges with labels.

- The word 'This' is a determiner (**det**) of 'virus'.
- 'virus' is the direct object (**nsubj**) of 'affects'.
- 'affects' is a verb and the head of the clause.
- 'the' is a determiner (**det**) of 'body'.
- 'body' is the possessor (**poss**) of 'defense'.
- 'system' is the direct object (**dobj**) of 'affects'.
- 'system' is the direct object (**nn**) of 'defense'.
- 'defense' is a noun and the head of the phrase.

### Word vectors

<system, dobj, affects> ...

Corpus-derived ternary data can also be mapped onto a 2-way matrix

Pawan Goyal (IIT Kharagpur)

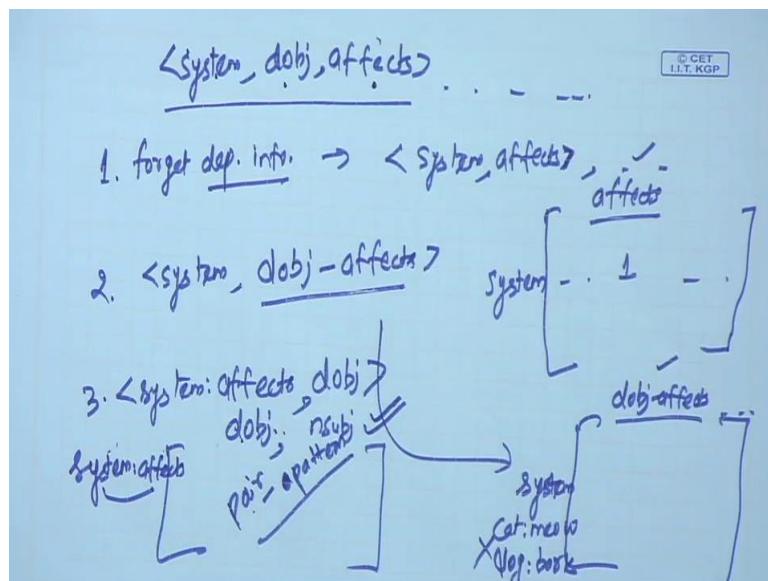
Distributional Semantics: Applications, Structure

Week 7, Lecture 3

12 / 15

What will I do? So, let us see I have a sentence the virus affects the body's defense system and you get the dependency parse, now from the dependency parse I can extract the various triplets like system det object affects body possessive system. So, these are the various triplets I can extract from a dependency graph. So, I will have things like this system det object affects and so on, now how do I use those for my structured model? So, let us try to have a look.

(Refer Slide Time: 29:00)



We will have words like system d o b j and affects yes and I have many such pairs many such topples now how do I use those for my distributional semantics. So, there are actually many ways you can do this. So, one simple way would be just forget dependency information so; that means, you will have system affects and so on. So, that is you going back to your earlier relation where you are not using the instruction. So, this you can again convert into that kind of model the word system is here and what is the co occurrence with the word affects and you will have 1 and so on there is one way. So, where you forget the dependency information, but that is not what you wanted right you wanted this information for some reason. So, another option could be I combine dependency information with the context. So, my context is now structured. So, that is I will have system and my context is det object of the verbs affects. So, these are my context now.

Then I can represent system in this context det object of affects det object of other verbs and subject of these verbs and so on. So, now, you see immediately my dimensions are different from here I had only words here, I had verbs and some relation together, but I am going back to the same sort of matrix format and still I cannot use this dog is to bark and cat is to meow, I cannot do it here, now what is some other sort of representation from here that you can gather?

Other could be I combine these 2 in my rows, this become my target and this come become my context. So, that would be third would be. So, like system and affects coming my target and det object is my context. So, now, we can talk about my pair matrix. So, I have systems and affects these are my pairs and pattern right now only dependency relation det dependency relation det object and subject and so on and this is the general framework pair pattern now here we are talking about only very simple paths of length 1 what is the det relation between these 2 words you are free to choose a higher order path it can be there is a path of length det object n subject and so on and use that at your patterns here.

That can help you to capture all these things like X solves Y. So, you can also use what are the different words that occur in between these 2 words, if there are some other words occurring that can be your one of the pattern. So, here you have complete freedom of choosing, what are your patterns? So, these are simple dependency graph for simplicity, but you can choose any other patterns that you would like these patterns come from may be dependency graph come from the word co occurrence how what are the words that are

occurring between these words and so on and then you can compute similarity between various pairs.

(Refer Slide Time: 33:02)

### 2-way matrix

<system, dobj, affects>  
 <virus, nsubj, affects>

*The dependency information can be dropped*

- <system, dobj, affects> ⇒ <system, affects>
- <virus, nsubj, affects> ⇒ <virus, affects>

*Link and one word can be concatenated and treated as attributes*

- virus={nsubj-affects:0.05,...},
- system={dobj-affects:0.03,...}



Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structures      Week 7, Lecture 3      13 / 15

(Refer Slide Time: 33:13)

### Structured DSMs for Selectional Preferences

*Selectional Preferences for Verbs*

Most verbs prefer arguments of a particular type. This regularity is known as selectional preference.

- From a parsed corpus, noun vectors are calculated as shown for 'virus' and 'system'.

	obj-carry	obj-buy	obj-drive	obj-eat	obj-store	sub-fly	...
car	0.1	0.4	0.8	0.02	0.2	0.05	...
vegetable	0.3	0.5	0	0.6	0.3	0.05	...
biscuit	0.4	0.4	0	0.5	0.4	0.02	...
...	...	...	...	...	...	...	...



Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structures      Week 7, Lecture 3      14 / 15

Now, quickly we have this data and I can use that to get this information or this, the other sort of information or the third sort of the information that we saw. Now let us see one simple application like how do we use that to find out selection preferences of the verbs, now what do I mean by this selection preferences? Now different verbs for their different argument like

object, subject, they prefer a particular type of noun and this information is useful in many task like dependency parsing.

For eat I want to know that eat will prefer only certain sort of nouns as objects you can eat only very very specific things. So, like, so how do we compute these selection preferences for verbs? So, we have seen that that from a parsed corpus I can compute these vectors like virus and system I can put them in this space how many times car occurs as object of carry. So, you are carrying car, you are buying car, you are driving car, you are eating car and you are storing car and flying car and so on.

What are the number of times and this is some normalized values vegetables how many times are they carried bought eaten stored and so on. So, this you can compute from the corpus once you have the parse. So, this gives you some sort of representation that what kind of objects come into that can be used, what kind of words that can come as object of the verb buy, what kind of words will come as object of the word verb drive and so on.

But suppose you want to build a prototype that is in general what kind of words can come as object of the verb eat from the corpus here 1 simple thing, I can do, I can find out which words has have a high value. So, you know vegetables can be eaten biscuits can be eaten. So, they have a high value, but suppose I want 200 prototypes for a new the words that is not occur in the corpus how likely it is to come as object of the verb eat. So, what will I do? So, what is the simple method? So, I would say let me find out what are the words that are having a high weight in this dimension. So, you know vegetables biscuits fruits etcetera will have a high weight in this dimension, now my hypothesis will be words that are similar to these words like vegetable biscuits can also be eat now how do I find out words that are similar to vegetables biscuits and so on.

For that I will build a prototype that is what are these words? So, these are words that can be carried bought stored right so; that means, any other words set can also be carried, bought, stored, might also be eaten. So, then I will find out all the other words that have high weights in these dimensions other than eat because whatever is having high dimension eat I can easily capture here, but what are the other words that are having high dimension in all these high weight in all these other dimensions that also can become my prototype.

(Refer Slide Time: 36:27)

**Structured DSMs for Selectional Preferences**

**Selectional Preferences**

- Suppose we want to compute the selectional preferences of the nouns as object of verb 'eat'.
- $n$  nouns having highest weight in the dimension 'obj-eat' are selected, let  $\{\text{vegetable, biscuit, ...}\}$  be the set of these  $n$  nouns.
- The complete vectors of these  $n$  nouns are used to obtain an 'object prototype' of the verb.
- 'object prototype' will indicate various attributes such as these nouns can be consumed, bought, carried, stored etc.
- Similarity of a noun to this 'object prototype' is used to denote the plausibility of that noun being an object of verb 'eat'.

Pawan Goyal (IIT Kharagpur)      Distributional Semantics: Applications, Structures      Week 7, Lecture 3      15 / 15

What will we do? Suppose I want to compute the selection preferences of the nouns as object of the verb eat I will take some top  $n$  words like vegetables biscuit that have high weight in this dimension of object eat then I take the complete vectors. So, that is of all these top nouns. So, this will words like that can be consumed bought carried stored etcetera now this becomes my object prototype now given any noun try to match it with this with this object prototype and you will see that how likely it is to come as a object of the verb eat and that is the generic method of finding selection preferences of an word noun for any word noun to come as a object or subject for verb.

So, we talked about, how do you extend your distribution semantic method for using structured models? A lot of work again a lot of research has gone to this domain. You have already touched the basic (Refer Time: 37:26) and I hope if you need this idea you on your own, you can think about how do you use different sort of interesting context, interesting targets and solve various problems.

So, in the next lecture, we will talk about another very import important interesting idea that is what are word vectors? So, and word embedding and how do we obtain them from the corpus and what are different tasks they can be used on? So, I will see you in the next lecture.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute Technology, Kharagpur**

**Lecture - 35**  
**Word Embeddings - Part I**

(Refer Slide Time: 00:36)

The slide has a blue header bar with the title "Word Vectors". Below it is a white content area. At the top of the content area, there is a bulleted list:

- At one level, it is simply a vector of weights.
- In a simple 1-of-N (or 'one-hot') encoding every element in the vector is associated with a word in the vocabulary.
- The encoding of a given word is simply the vector in which the corresponding element is set to one, and all other elements are zero.

Below this list is a section titled "One-hot representation" with a light purple background. It shows two vectors:

motel [0 0 0 0 0 0 0 0 0 1 0 0 0 0] AND  
hotel [0 0 0 0 0 0 0 1 0 0 0 0 0 0] =

At the bottom of the slide, there is a footer bar with the text "Pawan Goyal (IIT Kharagpur)", "Word Embeddings - Part I", "Week 7, Lec 1", and a small circular profile picture of the professor. There is also a navigation bar with icons for back, forward, and search.

Welcome back. So, we are talking about distributions semantics. So, in this fourth lecture and the fifth lecture of this week, we will talk about another very interesting idea that is word embeddings that come from distribution semantics. So, what are word embeddings? So, when I am talking about word embeddings, so they are like word vectors. So, the word vectors are nothing but simple vectors of vectors of weight. So, we talked about these factors. So, you are having some dimensions and they are representation words in this set of dimensions and their various weights.

So, now we also talked about this one-hot encoding that is among the n-dimensions only one dimension is 1, everything else as 0; and these dimension might correspond to the index of the particular word that I wanted to be set. Now, and we saw that if I if we are using one hot encoding I cannot compute the similarity among words. So, if I motel and hotel, there will have one at different places, and if I compute similarity, if I do an AND, it will be 0. So, this is not conducive to for semantic similarity between words.

(Refer Slide Time: 01:37)

### Word Vectors - One-hot Encoding

- Suppose our vocabulary has only five words: King, Queen, Man, Woman, and Child.
- We could encode the word 'Queen' as:

Pawan Goyal (IIT Kharagpur) Word Embeddings - Part I Week 7, Lecture 4 / 19

So, now let us take a simple example. So, have a vocabulary that contains only five words king, queen, man, woman and child. So, I have vectors in five dimension. So, I can encode queen like this. So, queen has a weight of 1 in the second dimension that correspond to the index of queen and it has weight 0 and other dimensions. And I cannot compute the how similar king and queen are, how similar queen and child are by using this method.

(Refer Slide Time: 02:12)

### Limitations of One-hot encoding

**Word vectors are not comparable**

Using such an encoding, there is no meaningful comparison we can make between word vectors other than equality testing.

Pawan Goyal (IIT Kharagpur) Word Embeddings - Part I Week 7, Lecture 4 / 19

So, we cannot make any meaningful comparison. We can only find if these two words are same, so that is not very interesting.

(Refer Slide Time: 02:21)

**Word2Vec – A distributed representation**

**Distributional representation – word embedding?**

Any word  $w_i$  in the corpus is given a distributional representation by an embedding

$$w_i \in R^d$$

i.e., a  $d$ -dimensional vector, which is mostly learnt!

$$\text{linguistics} = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}$$

Pawan Goyal (IIT Kharagpur) Word Embeddings - Part I Week 7, Lecture 4 5/19

So, what happens in the distributional representation or so we also called it word embeddings that any word  $w_i$  in the corpus is given a distributed representation by an embeddings. So, I have a fix dimension like  $d$ -dimensional vector, and I represent each word in these  $d$ -dimensions. And I will give them various weights and these weights are to be learnt by some method, and we will talk about what will be the method by which I will learn these weights. So, idea is that each word in my vocabulary, I will try to represent them in some fix dimensions  $d$ . So, this is one idea. So, like I have the word linguistics, and I can denote this word in some fix dimension here  $d$  is 8. So, there are 8 dimensions. So, it has different weights in different dimensions. And similarly all the words will be represented similarly they have different weights in these  $d$ -dimensions.

(Refer Slide Time: 03:26)

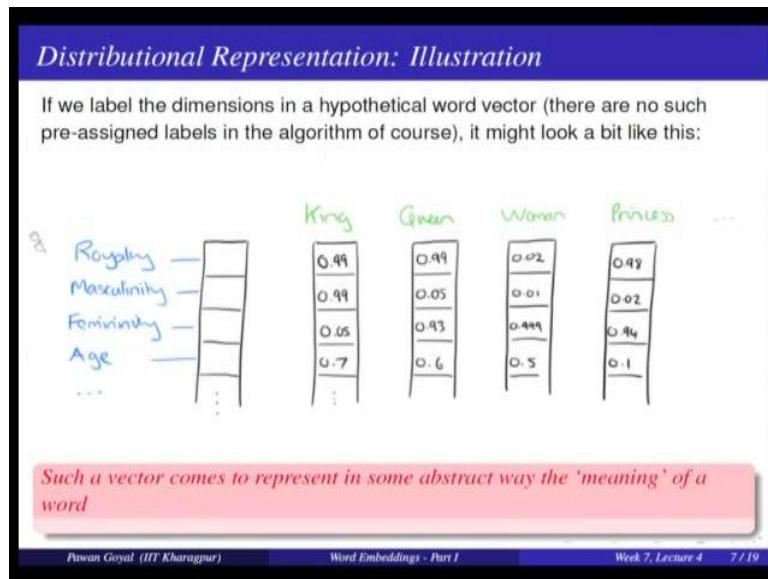
The slide has a blue header bar with the title 'Distributional Representation'. The main content area contains a list of bullet points:

- Take a vector with several hundred dimensions (say 1000).
- Each word is represented by a distribution of weights across those elements.
- So instead of a one-to-one mapping between an element in the vector and a word, the representation of a word is spread across all of the elements in the vector, and
- Each element in the vector contributes to the definition of many words.

At the bottom of the slide, there is a footer bar with the following text: 'Ptwan Goyal (IIT Kharagpur)', 'Word Embeddings - Part I', 'Week 7, Lecture 4', and '6 / 19'.

So, what is my distributional representation? So, take a vector with several hundred dimensions, so it can be 100, 50, 300, 1000. Now, each word is represented by a distribution of weights across these elements. So, each vector is nothing but a distribution of weights across these dimensions. So, what will happen? So, instead of a one-to-one mapping between an element in the vector and a word, you have a distributed representation of each word; and by using that you can capture similarity. So, whether two words are similar or not. If they have similar weights in many dimensions, then they will be similar and that is how I can capture similarity between words. So, we can say that each element in my vector or each dimension might contribute to the definition of multiple words.

(Refer Slide Time: 04:24)



So, what the dimensions might indicate this is just in illustration this is not a literal meaning, so that is what can you can help you to understand what are these dimensions. So, suppose all the dimensions in my distributional representation I can label by some hypothetical labels. So, my algorithm may not have some labels some very good labels like that it may not be possible to do that even manually, but this is simply for understanding that. Assume that so there are  $d$  dimensions assumed that you can assign a label to these some topic some concept like here my dimensions can be like royalty, masculine, feminine, age and so on, these are my dimensions. And assume that the weights that each for is have in this dimensions correspond to how much that word is closer to that concept.

So, for example, the word king word how much it is closer concept of royalty. So, this king is very close to concept of royalty it has a high weight in this dimension 0.99, it is very close to masculine, so it is has a high weight 0.99, but very small weight in feminine. Age suppose that a high weight means, it is elder, so it is 0.7. Similarly, for queen what will happen royalty will get a high weight yes, but masculine and feminine will be just opposite; and age can be again 3.6. Now, take women and princes women will have very low weight in the in the case of royalty high weight in the case of feminine. And princess will have a high weight in the case of royalty and high weight in the case of feminine and very low weight in the case of age. And this is simply for illustration.

So, idea is that I am trying to represent all my words instead of fixed dimensions. And these dimensions are latent, they may correspond to certain concepts a combination of concepts and so on that we may not know that the algorithm also does not assign. But we would assume that these dimensions correspond to some concepts, and now each word can be written as a distribution among these concepts. And then I can measure two words based on how much similar they are on various concepts, this is the idea. The main question is how do I capture this representation that how do I represent different words in this fixed dimensions. So, now we can see that such a vector is representing the meaning in some abstract manner.

(Refer Slide Time: 07:07)

The slide has a blue header bar with the title "Word Embeddings". The main content area contains the following text:

- $d$  typically in the range 50 to 1000
- Similar words should have similar embeddings

A pink callout box highlights the text: "SVD can also be thought of as an embedding method".

The footer of the slide includes navigation icons and text: "Pawan Goyal (IIT Kharagpur)", "Word Embeddings - Part I", "Week 7, Lecture 4", and "8/7/19".

So, now what is my dimension size? So, my dimension-d can be if we start from 50 up to 1000. And the key focus is that the words that are similar in meaning, they should have similar embeddings or similar representation. Now, if you know about SVD singular value decomposition that is also some sort of embedding method that converts the vectors in some low dimensions. So, I will encourage that you read about singular value decomposition, also another name for that is latent semantic indexing, but I will just give you very briefly what is the idea of SVD.

(Refer Slide Time: 07:50)

The image shows handwritten mathematical notes on a light blue background. At the top, there is a matrix equation  $A = U \Sigma V^T$ . To the left of the equation, the text "SVD" is written diagonally, followed by "embeds". Above the matrix  $A$ , there is a bracket labeled "500k" indicating its size. To the right of the matrix, the text "hyper-dim" is written. Below the equation, there is a bracket under  $\Sigma$  labeled "500k" with the text "↓ k-rank approx." underneath. To the right of  $\Sigma$ , the text "Σ - diagonal matrix" is written. Below the equation, there is another bracket under  $V^T$  labeled "Ak" with the text "singular value" underneath.

So, we talked about this co occurrence matrices. So, this is my matrix  $A$ . So, these are the target words, these are contest words. And this matrix has certain entries. Now, what is one problem is matrix this might be very high-dimensional. So, each word might have a say 500 k-dimensional representation. So, what this is co occurrence with different words and these words can be age number of words can be age 500 k and even more in some cases. So, now I want to give a low dimension representation a distribution representation. So, what will the idea? So, I will use the theorem that any matrix say can be written as  $U \Sigma V^T$ , and there are certain properties of these  $U$  and  $V$ , but this  $\Sigma$  is a diagonal matrix, and the entities are singular values. So, you can get about it what is this matrix particular decomposition for singular values.

Now, once you have this matrix in this format, this is same matrix. So, idea is that you take a  $k$  rank approximation so that means, you have singular values they are arranged in the decreasing order and you take only the top case singular values, and you do that for all this  $U$ ,  $\Sigma$ , and  $V$ . So, we have taking only top  $k$  entries. So, only the first  $k$  entries of  $U$  corresponding to top  $k$  singular values that is  $U_k$ ,  $\Sigma_k$ ,  $V_k^T$  transports and this is my  $k$  rank approximation of my matrix  $A$ . So, now this  $U$  will denote my low dimensional representation. So, earlier  $U$  might be in the same dimensions,  $U$  can be again in 500 k dimension 5,00,000, but suppose you are trying to pick only 100 you will take only first hundred dimensions from here, and this will be a low dimensional representation for  $U$ ,  $V$

and so on. So, SVD is also one sort of embedding. So, each word you can embed some k-dimensions by taking the top case singular values only.

But we are not talking about SVD here; we will talk about the word vector method for computing this representation. So, now before I discuss what are the different tasks sequence do with these word vectors, oh sorry how do you compute these word vectors, let us see why they are found to be very interested in this domain and what are the different tasks they have been used in.

(Refer Slide Time: 10:53)

**Reasoning with Word Vectors**

- It has been found that the learned word representations in fact capture meaningful syntactic and semantic regularities in a very simple way.
- Specifically, the regularities are observed as constant vector offsets between pairs of words sharing a particular relationship.

**Case of Singular-Plural Relations:**  
If we denote the vector for word  $i$  as  $x_i$ , and focus on the singular/plural relation, we observe that

$$x_{apple} - x_{apples} \approx x_{car} - x_{cars} \approx x_{family} - x_{families} \approx x_{cat} - x_{cats}$$

and so on.

Navigation icons: back, forward, search, etc.

Page footer: Praveen Dandekar (IIT Kharagpur), Word Embeddings - Part I, Week 2, Lecture 4, 9/219

So, what is we found that these representations are capturing some meaningful syntactic and semantic regularities in a very, very simple manner. So, what is that? So that is we can use the vector offsets to talk about relation between words and how much two words are similar compare to the other pair of words. So, for example, I want to capture the singular plural relationship between words like car and cars, boy and boys, bag and bags and so on. And suppose I do not know what are singular and plural words. So, can I capture that using word representation?

So, what does we found suppose that for each vector  $x_i$  for each word  $i$ , you have vector  $x_i$ . So, we can take the vector offsets and they will be coming out to be similar for singular to plural pair, so that is if I try to compute  $x_{apple} - x_{apples}$  that is coming out to be very close to  $x_{car} - x_{cars}$ , similarly to  $x_{family} - x_{families}$  and so on. That is I compute the vectors of these and if I take the vector offset  $x_{apple} - x_{apples}$  has similar

offsets as car minus cars as family minus families, and this is very, very interesting this is not something for which this vectors for trained for, but they are capturing this regularity very nicely.

(Refer Slide Time: 12:23)

*Reasoning with Word Vectors*

Perhaps more surprisingly, we find that this is also the case for a variety of semantic relations.

*Good at answering analogy questions*

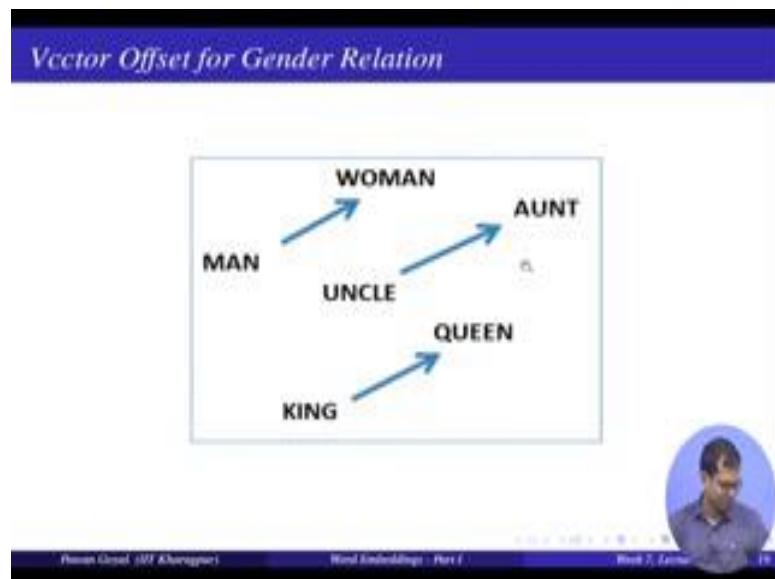
a is to b, as c is to ?  
man is to woman as uncle is to ? (aunt)

A simple vector offset method based on cosine distance shows the relation.

Previous (Page 107) Next (Page 109)  
Word Embedding - Part I  
Word 2, Lecture 8 (Page 109)

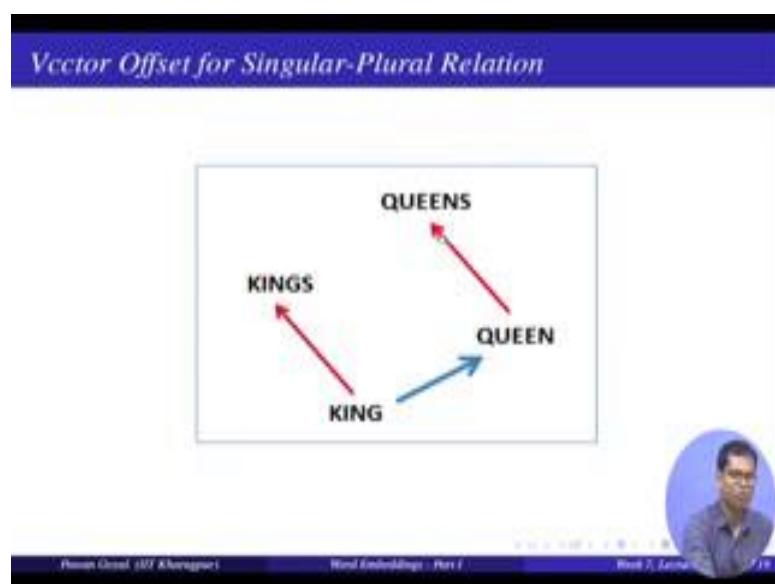
So, this would be like and because they capturing such regularities they can be used for various reasoning task like analogy task a is to b as c is to what. So, like man is to woman as uncle is to and you will answer aunt, but can my model answer that man I have given man woman pair, and for the next pair the first word is uncle, what be the next word, can you find out aunt. And that is what the word vectors have you found to be very useful in; they can predict these analogy sort of task. And how do we do that they just use the simple vector offset method.

(Refer Slide Time: 13:03)



So, let us see one example. So, here so this is the idea. So, I have my vectors denoted in a two-dimensional plane. So, how do you convert any say 100 dimensional vectors to two-dimensional representation, you can use principle component analysis PCA or some other methods to project them into some lower dimension, so that is been done here, two-dimensional representation. So, what you are seeing here? So, that outside would be woman and man that is similar to what has been observed in uncle and aunt, and king and queen and this is a very nice regularity.

(Refer Slide Time: 13:39)



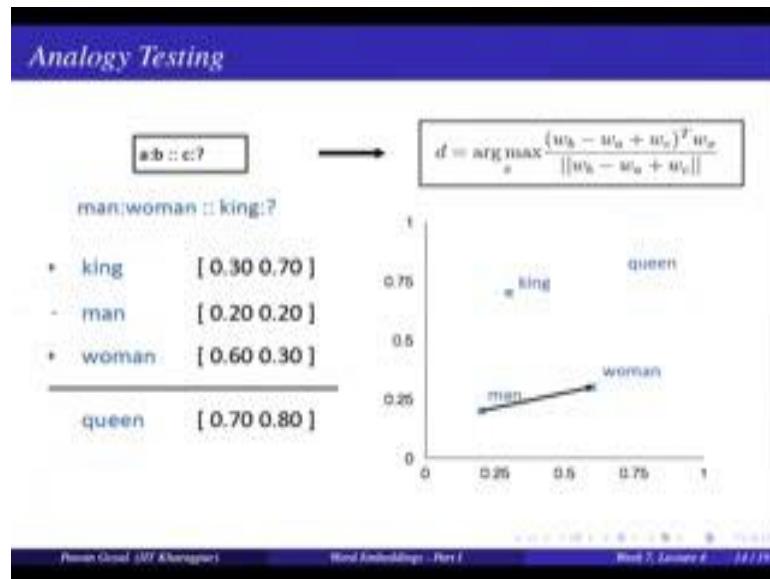
Similarly, here for singular plural king to kings and queen to queens, they are having similar offsets.

(Refer Slide Time: 13:48)

Encoding Other Dimensions of Similarity			
Analogy Testing			
Relationship	Example 1	Example 2	Example 3
France : Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big : bigger	small : larger	cold : colder	quick : quicker
Miami : Florida	Baltimore : Maryland	Dallas : Texas	Kona : Hawaii
Einstein : scientist	Mozart : mathematician	Mozart : violinist	Picasso : painter
Sarkozy : France	Berlusconi : Italy	Merkel : Germany	Koizumi : Japan
copper : Cu	iron : Zn	gold : Au	uranium : plutonium
Berlusconi : Silvio	Sarkozy : Nicolas	Putin : Medvedev	Obama : Barack
Microsoft : Windows	Google : Android	IBM : Linux	Apple : iPhone
Microsoft : Bill Gates	Google : Yahoo	IBM : McNealy	Apple : Jobs
Japan : sushi	Germany : beerfest	France : tapas	USA : pizza

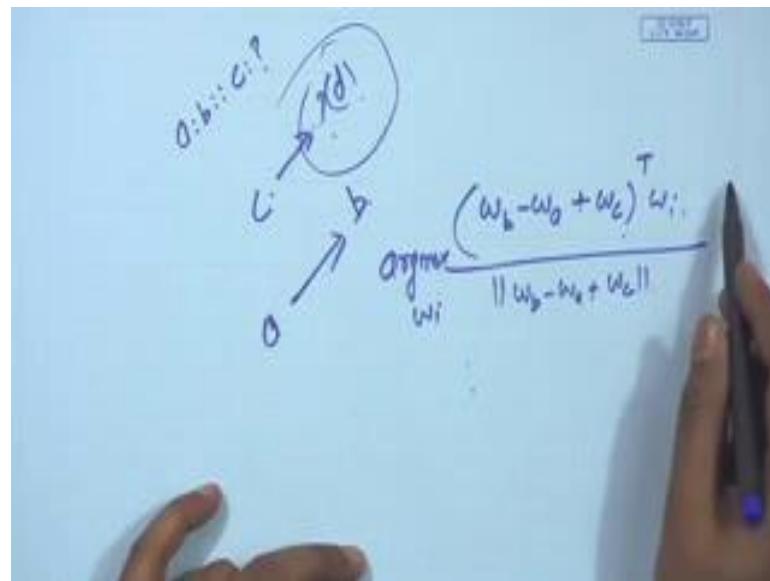
So, we can use that for analogy testing. So, what is the analogy testing task you are given a pair with a certain relation like France and Paris. What is the relation? So Paris is the capital of France. Now, we are given various examples and you have to find out which of this examples exhibit the same relation, so that is whether Italy, Rome has a relation of capital and country; Japan, Tokyo and Florida Tallahassee which of these has the same relation. So, can my vector representation capture this? So, a given one example can you find out the other example. And so similarly for big, bigger can you find out this small smaller, cold colder, quick quicker, and Miami Florida can you find out other examples so on.

(Refer Slide Time: 14:42)



And how do they do that. So, I have this task a is to b as c to what. So, example is man is to woman as king is to sorry man is to woman as king is to what. And how will they do that. So, simple idea is take the vector offsets between woman and man, and add it to king. So, find out woman minus man, add it to king, but how do you find out the word queen, in general it may not be the exact match. So, how do you find out what words are coming closer.

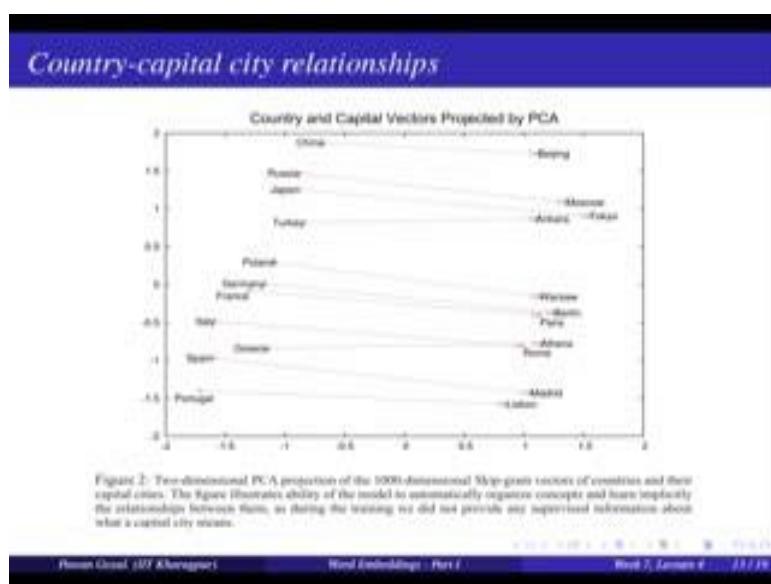
(Refer Slide Time: 15:20)



So, this is the idea. So, I have a is to b as c to what. So, what will happen in my vector representation a, b and c, I want to find out what is the word d. So, what will I do, I have the

vectors of each of the words. So, I will find out the offsets  $w_b - w_a$  added to  $w_c$ . Now, I am trying to find out which vectors are similar to this. I write it this and where what are the vectors are similar to this one. So, I will just find out similarity of that to all the words  $w_i$  in my corpus and I can also normalize it  $w_b - w_a + w_c$ , so that is a normalize all this is not very important and we take the argmax over all  $w_i$  in my corpus. So, all words are vectors. So, I find out which words are coming closer in this space. So, what is the closest words to by when I add this offsets to this word and that is my answer, for example, if you do this here, so will find that the word queen comes up.

(Refer Slide Time: 16:45)



And this has been shown in many different cases like country and capital vectors, China-Beijing, Russia-Moscow, Japan-Tokyo, and what you are seeing here the offsets between the vectors are very, very regular in all these cases. And this is just coming out from the word vectors.

(Refer Slide Time: 17:06)

Newspapers			
New York	New York Times	Baltimore	Baltimore Sun
San Jose	San Jose Mercury News	Cincinnati	Cincinnati Enquirer
NHL Teams			
Boston	Boston Bruins	Montreal	Montreal Canadiens
Phoenix	Phoenix Coyotes	Nashville	Nashville Predators
NBA Teams			
Detroit	Detroit Pistons	Toronto	Toronto Raptors
Oakland	Golden State Warriors	Memphis	Memphis Grizzlies
Airlines			
Austria	Austrian Airlines	Spain	Spanair
Belgium	Brussels Airlines	Greece	Aegean Airlines
Company executives			
Steve Ballmer	Microsoft	Larry Page	Google
Samuel J. Palmisano	IBM	Werner Vogels	Amazon

Table 2: Examples of the analogical reasoning task for phrases (the full test set has 3218 examples). The goal is to compute the fourth phrase using the first three. Our best model achieved an accuracy of 72% on this dataset.

Pawan Goyal (IIT Kharagpur) Word Embeddings - Part I Week 7, Lecture 4 16 / 19

And more questions like newspapers. So, New York and New York Times (Refer Time: 17:12) Baltimore and Baltimore Sun, San Jose San Jose Mercury News and so on; various NBA teams Detroit Detroit pistons and so on; airlines - Austria Austrian airlines, Belgium Brussels airlines and company executives. So, what do you see it is capturing a generic relation also? In general, any relation that can hold between two words, so if you are finding out two words with one relation you can find out some many other words that have the same relation by simple this vector offsets method. So, find out vector offset between pairs, is it similar to the vector offsets of my initial example.

So, this is the problem we also tackled in the case of structured models of distribution semantics that is we were making the pair pattern matrix and then capturing the co occurrences. In the case of word vectors, even though you did not start by pairs and patterns, even though you found out only with the word embeddings word vectors this helped further in doing this task also, and this was very one of the very interesting aspects.

(Refer Slide Time: 18:27)

*Element Wise Addition*

We can also use element-wise addition of vector elements to ask questions such as 'German + airlines' and by looking at the closest tokens to the composite vector come up with impressive answers:

German + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
korona	Hanoi	airline Lufthansa	Moscow	Julieta Binoche
Check currency	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zoty	Viet Nam	flag carrier Lufthansa	igorsk	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile Dele

Table 5: Vector compositionality using element-wise addition. Four closest tokens to the sum of two vectors are shown, using the best Skip-gram model.



Pavan Chital (M.Tech) Word Embedding - Part I West P. Lecture 1

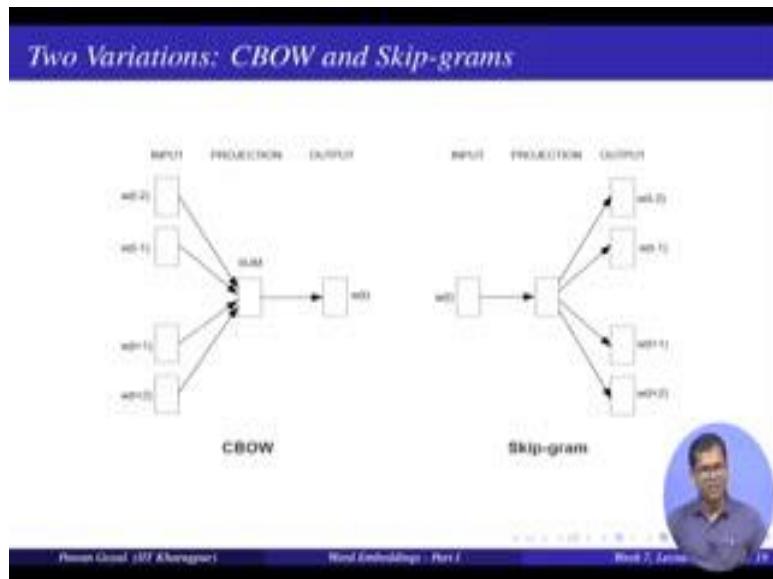
Similarly, here we can do element-wise addition. So, suppose you have the embedding for a German, you have embedding for airlines and if you add these two embeddings, and find out words that are coming closer to the new vector now. And you find out some very interesting words coming up like here check plus currency, and you find out word that very close, Vietnam plus capital again words setup very close German plus airlines you find Lufthansa airline and Lufthansa a carrier so on, Russian plus river you find words like Moscow French plus actress and you find some French actress. And this was again some interesting aspect. So, you can have embedding of two words and you add these gather if find something that is some sort of composition of these term. So, it is not a generic method. So, this was coming out in some cases may not be true for all the cases, but even coming out in some cases for what in interesting observation.

(Refer Slide Time: 19:22)

*Instead of capturing co-occurrence counts directly, predict (using) surrounding words of every word.*  
Code as well as word-vectors: <https://code.google.com/p/word2vec/>

So, now how do we capture these word vectors, how do we compute this word vectors. Now, basic idea is we will again go back to the co occurrences, we will trying to use co occurrences. But instead of counting the co occurrences from a corpus, what we will say I am given a sentence a word is there, and the contest is there, try to predict the contest from the word or the word from the contest. And if you not able to predict, try to update your weights, and this will be the idea. It starts with some initial vectors, using those vectors try to predict. From the contest, what to do the targets, after the target what will be the contest if it is not matching update your vectors. So, all the codes as well as the learned vectors are available here. So, you can actually download these and try to use them for many of your task also, but we will discuss how what are this word vectors in how to they come.

(Refer Slide Time: 20:26)



So, in general, there are two different variation of this models, so that tried to capture these word embeddings, and they are called CBOW for continues back of words model and skip-gram models. And let me just quickly explain what are these and we will discuss in details in the next lecture. So, in continuous back of words model, what you are doing, so, we are taking the contest. So, you are focusing on the current word  $w_t$ , you are taking the previous words. So, it can be actually any number of previous words and to next words and you are trying to use those to predict the center word. So, using the contest predict the target word or the center word.

In the skip-gram model, what you are doing you are using the center word and trying to predict the context words. So, there are two different ways of learning these embeddings. So, idea would be I will start with some initial vectors, now trying using the context vectors, I will try to predict what is my center vector. If I am not finding the match I will update my different vectors center as well as the context. And I will keep on doing that until I am able to predict this some high confidence or I am converging at certain point my vectors are not changing. And these vectors that I am learning by this method will be my word vectors and I do that slightly different in both continuous back of words model and skip-gram model.

So, in the next lecture, what we will do, we will discuss in detail how do we learn these vectors.

Thank you.

**Natural Language Processing**  
Prof. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 36**  
**Word Embeddings – Part II**

Hello everyone, welcome to the fifth lecture of this week. So we are talking about word embeddings. So in the last lecture we discussed the what is the different ideas behind using word embeddings, what can be some interpretation given to the different dimensions here and what are different tasks where you can use these word embeddings. But we do not go through the learning part, how do we actually obtained these word embeddings for different words. So today in this lecture we will try to discuss in detail how do we obtain these embeddings.

(Refer Slide Time: 00:53)

**CBOW**

- Consider a piece of prose such as:  
"The recently introduced continuous Skip-gram model is an efficient method for learning high-quality distributed vector representations that capture a large number of syntactic and semantic word relationships."
- Imagine a sliding window over the text, that includes the central word currently in focus, together with the four words that precede it, and the four words that follow it:

...an efficient method for learning high quality distributed vector ...

content ↑ context  
focus word

Pawan Goyal (IIT Kharagpur) Word Embeddings - Part II Week 7, Lecture 3 2 / 14

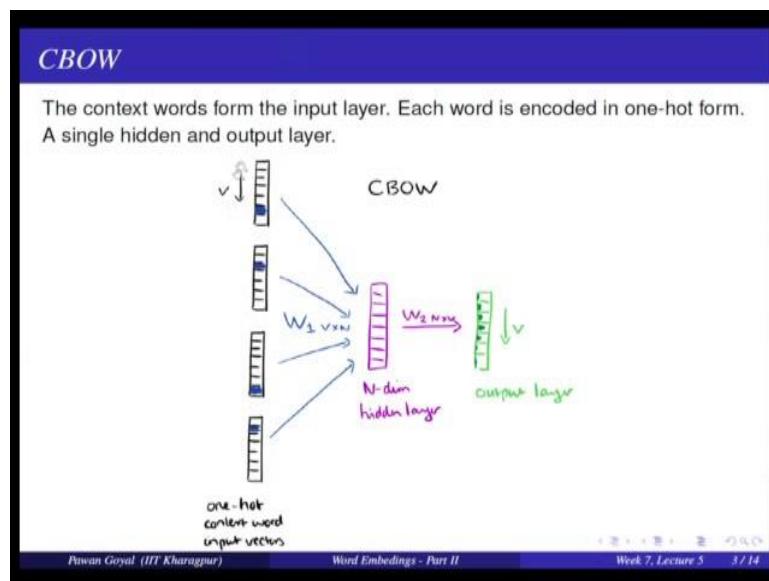
So in the last lecture we discussed that there are 2 different models for word embeddings. So that mean two main models one is continuous back away words model and another is skip gram model. What is the idea there? In continuous back of words model, using the neighboring words, I am trying to predict the center word. In a skip gram model using the center word I am trying to break the neighboring words.

So let us see what we do in CBOW model. So here let us say we have a prose like this. The recently introduced continuous skip gram model is an efficient method for learning

high quality distributed vector representation and so on. So that is the test data that I have so much and I am trying to learn representation for various words embeddings.

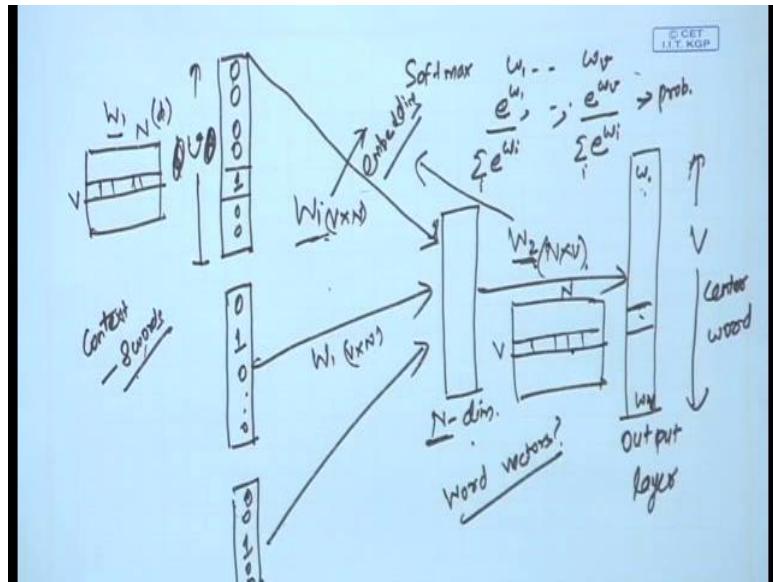
So at a time I will go through one particular window here. So I will focus on a word. So let us say I am focusing on one word like learning. So what I will do? I will take a window around that word. Now window size can vary and this can be a hyper parameter that you will choose. So you can choose a I will take forward before there and forward after that. So using these 8 words around this word learning, I will try to predict this word learning. So how do we do that? So imagine is sliding window over the text, that includes the central word and with 4 words that precede and the 4 words that follow it. Like here learning is my focus word, and there are 4 words before it and for words after that. And you can also call that as the context words. And usually the context words you are trying to break the focus word.

(Refer Slide Time: 02:40)



So this is some sort of network representation and how this learning will take place. So you are seeing here in the input you have, input you have various column vectors. So what do I mean by this? So what you are doing here you are putting in one hot encoding as the input. So I have this column vector of size  $v$ .

(Refer Slide Time: 03:09)



So the column of size  $V$  now, so  $V$  is the size of my vocabulary. I have that many words in my vocabulary. So I may not use this only  $V$ ,  $V$  is the size of my vocabulary. Now in your context you are heading having certain word that will be there in the vocabulary. So let us say this is the index of that word. So this input would be everything else a 0 and this will be 1. So one hot encoding of that word, so column vector; so like that in your context suppose you have here 8 words. So we will feed all these 8 one hot forms. So here it might be one here 0 here it might be one somewhere and everything else 0. That is your input that you are feeding. Now using this input then you are having a hidden layer this is  $N$  dimension. I remember this  $N$  will be important we will see what is this  $N$ .

So now the size  $V$ . Now I want to map these 2 this  $N$  dimensions. So I will have a weight matrix. Let us say  $w_1$  of dimension. So this is  $V$  cross 1 sorry. So this you can take it as 1 cross  $V$ . So you will have it as  $V$  cross  $N$ , so that the final representation is  $N$  dimensional now. So you will have the same weight matrix at each input. So you will get and then you can take the average of all these now from these  $N$  dimension you again go to your  $V$  dimension. This is you are output layer. So what will be the second weight here? This will be  $N$  cross  $V$ . See I starting with a  $V$  dimension going to  $N$  dimension and hidden and then going to the  $V$  dimension in the output now. So what you are doing here? So before going into what are the connections a network you are putting some input there are your context words. So suppose you are having 8 different one hot vectors and whatever happens here finally, you are trying to create your center word. This should

be your center word. So what will happen here? You will have various weights  $W_1, W_n$  some numbers you will  $W_V$  some  $V$  different numbers you will get.

And you want to ensure that the actual center word. Suppose this is my center word. This is the highest probability. And if it is not having the highest probability, you will try to modify the weights in your network. So that it gets a higher probability than what is getting right now.

So now let us see what do we mean by all these connections. So what would happen from here? So let us go back to the slides. So each word is encoded in one hot form that is here. We have a single hidden layer and an output layer. So you are having 2 different weight matrixes one is  $V \times N$  another  $N \times V$ . So from input you go to hidden from hidden you go to output. Output you are trying to predict the center word.

(Refer Slide Time: 07:15)

*CBOW: Training Objective*

- The training objective is to maximize the conditional probability of observing the actual output word (the focus word) given the input context words, with regard to the weights.
- In our example, given the input ("an", "efficient", "method", "for", "high", "quality", "distributed", "vector"), we want to maximize the probability of getting "learning" as the output.

Pawan Goyal (IIT Kharagpur)      Word Embeddings - Part II      Week 7, Lecture 5      4 / 14

So what is my training object is here? So you see I am getting some word, some numbers for different all my  $V$  words at the output layer. And this you can think of as the probability value. And you might have a question that how do I convert these numbers to probability we will see that. So assume that you having  $V$  different probability values, now you want to maximize the conditional probability of observing the actual output word given the input context word with regard to weights.

So you say that I want to maximize the probability that I will obtain the actual what that I saw in my input, and not any other word. So whatever probability I am getting that becomes my objective function. So I want to maximize this condition probability of output word given all my input words. And this probability will be expressed in terms of all these weights that I am having in my network. So in our example what would happen, I am giving this input all these 8 words an efficient method for high quality distributed vector, and I want to maximize the probability of getting learning as the output this is my center word.

(Refer Slide Time: 08:24)

**CBOW: Input to Hidden Layer**

Since our input vectors are one-hot, multiplying an input vector by the weight matrix  $W_1$  amounts to simply selecting a row from  $W_1$ .

$$\begin{bmatrix} \text{input} \\ 1 \times V \end{bmatrix} \begin{bmatrix} W_1 \\ V \times N \end{bmatrix} = \begin{bmatrix} \text{hidden} \\ 1 \times N \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{bmatrix} = \begin{bmatrix} e & f & g & h \end{bmatrix}$$

Given  $C$  input word vectors, the activation function for the hidden layer  $h$  amounts to simply summing the corresponding 'hot' rows in  $W_1$ , and divide by  $C$  to take their average.

Now what happens from input to the hidden layer? So input we are saying you are feeding one hot vectors. So if there are 8 different input words I am having 8 different one hot vectors. Now what do I mean by multi bank this one hot vector with my first weight matrix that is of dimension  $V$  cross  $N$ . So you can think of it like that. So you put using an input. So this is one word and having a corresponding element is one that that is my input word and you are multiplying it with  $V$  cross  $N$  weight matrix. So this operation is nothing like this nothing, but you are taking the corresponding row of this matrix. This matrix has  $V$  different rows. So these  $V$  rows you can think of as if corresponding to  $V$  different words in my vocabulary. So whenever you are feeding a one hot form of one word you are picking up that row and so you are doing it for 8 different words. So you are picking 8 different rows in my initial weight word weight for matrix, and then you are having given  $c$  input words. So activation function for the

hidden layer will be simply summing the corresponding hot rows. So I will pick up the hot rows here that correspond to the input words and divide by c, so that I have an average representation.

So now you understand what is going from input to hidden layer. You are taking c different one hot c different rows from your weight matrix and averaging those that is what goes in your hidden layer.

(Refer Slide Time: 09:59)

The slide has a blue header bar with the text "CBOW: Hidden to Output Layer". Below the header is a large white area containing a text box. The text box contains the following text: "From the hidden layer to the output layer, the second weight matrix  $W_2$  can be used to compute a score for each word in the vocabulary, and softmax can be used to obtain the posterior distribution of words." At the bottom of the slide is a video player interface. It includes a circular profile picture of a man, the name "Pawan Goyal (IIT Kharagpur)" on the left, and "Word Embeddings - Part II" and "Week 7, Lecture 1" on the right. There are also navigation icons for the video player.

Now, what happens from hidden to output layer? So remember we have a second weight matrix  $W_2$  that goes from hidden to output layer and its dimension is  $N \times V$ . So now, a hidden layer dimension is  $1 \times M$ . So if I multiply this  $1 \times M$  matrix with this  $N \times V$  matrix I will get a  $1 \times V$  matrix and that is my output layer. So this  $1 \times V$  matrix even think of as having weights for different words in my vocabulary. And I want to maximize weight for my center word. So from the hidden layer to output layer the second weight matrix  $W_2$  can be used to compute a score for each word in my vocabulary and now you can obtain the weights. How do we convert them to probability values and for that you use soft max? What is the idea of soft max?

So here you will get the weights  $W_1$  to  $W_V$ . They can be any real numbers. Now how do you convert that to probability distribution? So this is a simple idea that is in soft max. So in soft max what you do is you are given  $W_1$  to  $W_V$  and you want to convert that to a probability distribution. So you will see that I simply multiply all these by. So I

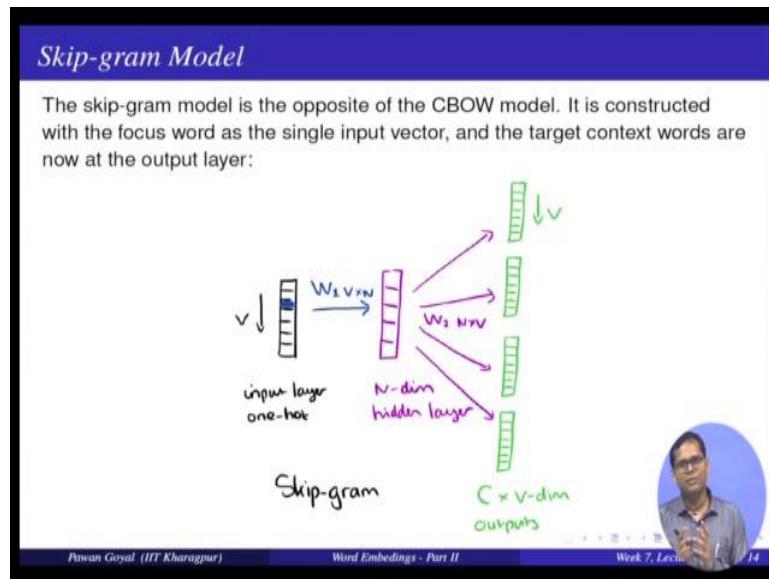
will put an exponent over these, so  $e$  to the power  $W_1$   $e$  to the power  $W_v$ . Now they are all positive numbers and then to convert them to probability I will just divide by summation over  $e W_i$  for all  $i$ .

So now this is this will sum off to 1 because this is normalize and these are my probability distribution. So I have these weights I use a soft max to convert them to probability values. And then I will try to maximize the probability of my actual center word. This will become my training objective and based on that I will learn my weights. So I will talk a bit about this learning problem in the when I go to the next model of escape ground, but suppose we have some way of learning these weights, and finally, I have the optimal set of weights  $W_1$  and  $W_2$ . Now where are the word vectors, what are your word vectors. Here I am feeding one hot vector, this is not being learned. This is the same. So where are my word vectors being learned now if you think about it. So what I am learning are the weight matter research. This is off  $V$  cross  $N$  this is  $N$  cross  $V$ . So I can take a transfer this also will become  $V$  cross  $N$ . So you can think of it as if for each word you are learning a  $N$  dimension representation. So after you learn this matrix these weight one and weight 2 will correspond to your word embeddings.

So  $W_1$  will be of size, So  $W_1$  will be of size  $V$  cross  $N$ . So you can take any vector  $i$  and get a  $N$  dimensional representation and the sign you can also think of as your  $d$ ,  $d$  dimensional representation. So for each word you are getting a  $d$  dimensional vector. Similarly, for  $W_2$  you will get similarly if you take a transpose you will get a  $V$  cross  $N$  representation. And in general what you do, you can take a, so you are getting 2 vectors: one from  $W_1$  from  $W_2$ . So we can finally, combine these 2 vectors you can either concatenate these vectors or the same word or some these over or taken average and that works fine. So this is the addition about what is the kind of network that you use for learning these embeddings. So these weights are nothing, but my embeddings. That I am learning. So this is about continuous back of words model. Now what will happen skip gram, in a skip gram model the network will slightly change. So now, I will feed only the center word here only one input vector, but I will predict multiple contexts words.

So now from input to hidden there are only one. So the only one input vector, but output there are multiple vector.

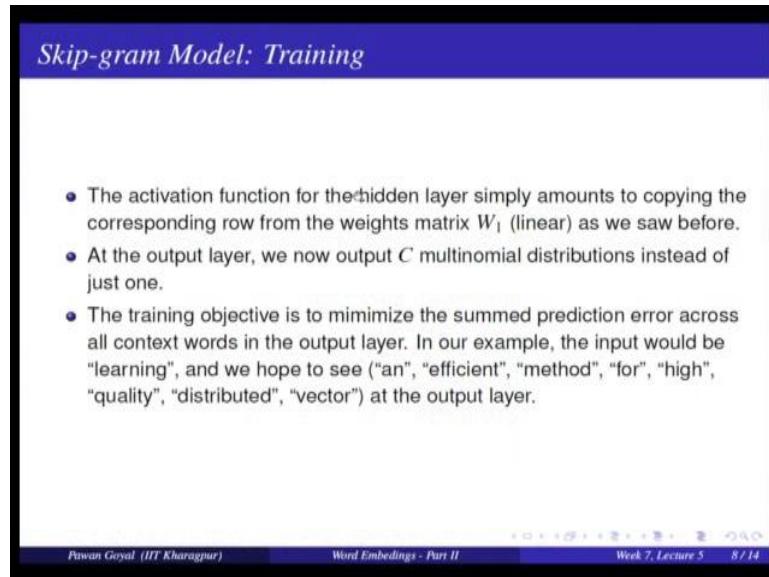
(Refer Slide Time: 14:41)



So let us look at the skip gram model. Skip gram model is the opposite of CBOW model. So you have the center word, as the single input vector and the target context words are at the output layer. This is my output layer. And you can now very easily correlate with what we did in CBOW. So the weights correspond to roughly the same idea.

So now let us formally define so we see how in the using the network we can think about the learning. How the learning will take place and how the weight updates will take place, but suppose you want to write it mathematically and how do we do that?

(Refer Slide Time: 15:19)



So yeah this is very analogous to what we find the case of CBOW. So here from input to hidden layer I am simply copying the row from the weight matrix  $W_1$ . So I am having only one input. So I will copy only one row and that will go to my hidden layer. Now at the output layer we will have see different distributions. And I will try to predict the see different context words using my hidden layer and objective is to maximize some prediction errors across all the context version my output layer. So you want to predict although. So I want to maximize the probability for observing my actual context words. So let us say we have a window, where I am having small  $c$  words in the left small  $c$  words in the right.

(Refer Slide Time: 16:20)

The image shows a handwritten derivation of the training objective function  $J(\theta)$  for a neural network language model. The derivation starts with the formula:

$$J(\theta) = \sum_t \sum_{-c \leq i \leq c, i \neq 0} \log p(w_{t+i} | w_t)$$

Below this, it shows the hidden layer representation  $v_{wI}$  as a vector of size  $n$ , with a note "Each word - 2 vectors". It also shows the output layer representation  $v_{wo}$  as a vector of size  $m$ , with a note "Output - 2 vectors".

The next part shows the calculation of the probability  $p(w_o | w_I)$  as:

$$p(w_o | w_I) = \frac{\exp(v_{wo}^T v_{wI})}{\sum_{w=1}^W (v_w^T v_{wI})}$$

With a note "(Network)". Below this, it shows the normalization condition:

$$\sum_{w_o} p(w_o | w_I) = 1$$

So what can be my training objective? It can be I am trying to maximize, suppose I take a log probability  $W_t$  plus a given  $W_t$ .  $W_t$  is the center word and I am going from minus  $c$  less than equal to  $j$  that is equal to  $c$ . Window of size  $c$  around  $j$  and  $j$  is not equal to 0. And this I can do for all possibility. All possible center words and this becomes my training objective.

(Refer Slide Time: 16:56)

The slide has a blue header bar with the text "Skip-gram Model". Below it is a purple box containing the word "Details". Inside the purple box, the text says "Predict surrounding words in a window of length  $c$  of each word" and "Objective Function: Maximize the log probability of any context word given the current center word:". Below this is the mathematical formula for the objective function:

$$J(\theta) = \frac{1}{T} \sum_{t=1}^T \sum_{-c \leq j \leq c, j \neq 0} \log p(w_{t+j} | w_t)$$

At the bottom of the slide, there is a circular profile picture of a man, the name "Pawan Goyal (IIT Kharagpur)", the title "Word Embeddings - Part II", the week "Week 7, Lecture 14", and a navigation bar.

$$J(\theta) = (1/T) \sum_{t=1}^T \sum_{-c \leq j \leq c, j \neq 0} \log p(w_{t+j} | w_t)$$

So we will see that. So I am predicting surrounding words in the window of length  $c$  of each. My objective function is I want to maximize the log probability of any context word given the current center word. So I am trying to maximize this probability. Probability of  $w_t + j$  given  $w_t$  and sum over all the possible context words and then I can sum it over all the possible words in my input, and this becomes my overall objective, so  $J(\theta)$  and  $\theta$  of all the parameters all the weight matrix  $W_1$  and  $W_2$  that I am trying to learn.

So now this is my objective function and I want to maximize that and by doing that I want to learn my parameter  $\theta$  and how do we do that.

(Refer Slide Time: 17:51)

For  $p(w_{t+j}|w_t)$  the simplest first formulation is

$$p(w_o|w_t) = \frac{\exp(v'_{wo}^T v_{wt})}{\sum_{w=1}^W \exp(v'_w^T v_{wt})}$$

where  $v$  and  $v'$  are "input" and "output" vector representations of  $w$  (so every word has two vectors)

Pawan Goyal (IIT Kharagpur) Word Embeddings - Part II Week 7, Lecture 5 10 / 14

$$p(w_o|w_t) = (\exp(v_{wo}^T v_{wt})) / \sum_{w=1}^W \exp(v_w^T v_{wt})$$

Firstly, let us see how will we compute these probabilities. Probability  $W_t$  plus  $j$  given  $W_t$ ; now this will be actually you have already seen that in the case of using the network but using which some mathematics can be shown that. So idea is that let us say I am having a probability for  $P(W_o)$  given  $W_I$ ,  $W_I$  is my context word, and  $W$  is the output word. That is the; I am sorry so I should say as the center word. And this is my output word these are all the context words. Let us say for a given context words how do we write it. And I am saying we can write it in this form. That is exponent  $V'$   $W_o$  Transpose  $V$   $W_I$  divide by sum over  $W$  is equal to one to capital  $W$   $V'$   $W$  transpose  $V$   $W_I$ . And these are formulation of this probability. And let us try to understand that. So whatever different is we have written here. So one thing you are seeing there is a  $V$  and there is a  $V'$ . So for each word you have 2 vectors. One is  $V$  another is  $V'$ . So  $V$  is for the so  $V$  is used only for the input center word. And  $V'$  is used for output.

So when I am trying to use the input word to predict the output word I will simply take a dot product of these. Now both are vector like column vectors. So  $V$   $W_I$  will be of this form, and  $V'$   $W$  will also be as a column vector. So how do I get a number by multiplying these 2? I will take a transpose of this and multiplied with this and that is what I am doing here.  $V'$   $W$  transpose times  $V$   $W_I$ . That gives me single number.

Now this number I want to convert to a probability value. So then I am using a softmax over that. Exponent over this number divide by I will do it for all the words in output. So that is why I have a summation over all the word,  $W$  is the total number of words in my vocabulary. This is nothing about a probability distribution, and you can see that summation  $P_{Wo}$  given  $WI$  for all words  $Wo$  will add to 1 because of this normalization factor. So this is added to word this is giving me a condition probability of  $Wo$  given  $WI$ . Now as a simple exercise you can also try to see how this number comes from the network that we talked about. So we had a networking interpretation. Now we have a simple matrix short of interpretation.

So how they correspond to each other? And they use the idea that word one sorry weights 1 and weighs 2. What are these? Weights weight 1 correspond to the input and way to correspond to output. Use this idea and see that by using the network also are you getting the same form of this probability by putting  $x$  softness over the output layer and you will you will see that you are actually getting the same form. So now, I have this formulation  $Wo$  given  $WI$  this particular formulation and I put it here for all the words in my context and this becomes my  $j \theta$ . This is my objective function.

Now what is my objective? I want to learn my  $V$  an  $V$  prime. So I will try to learn my  $V$  and  $V$  prime such that this is optimized and for that there is a simple way that is you can use getting descent algorithm. So you can try to take partial derivative of  $j \theta$  with respect to each of these parameters and update your weights accordingly. So this is the probability condition, probability distribution function that we have found and  $V$  and  $V$  prime are input and output vector representation of the same of the same word  $W$ . So every word has now 2 vectors.

(Refer Slide Time: 23:01)

### Parameters $\theta$

With  $d$ -dimensional words and  $V$  many words:

$$\theta = \begin{bmatrix} v_{aardvark} \\ v_a \\ \vdots \\ v_{zebra} \\ v'_{aardvark} \\ v'_a \\ \vdots \\ v'_{zebra} \end{bmatrix} \in \mathbb{R}^{2dV}$$

Pawan Goyal (IIT Kharagpur) Word Embeddings - Part II Week 7, Lecture 5 11 / 14

So suppose I have  $d$  dimensional vectors. So  $d$  is the dimension of my hidden layers in network terms or for each word what is the representation that we are using? We are using dimension representation. So I have many words. So what is the size of my parameter  $\theta$ ? So I have capital  $V$  words for each word I have been input vector and output vector. So there are  $2V$  vectors and each vector of dimension  $d$ . So I have  $2d$  times capital  $V$  that many parameters. This should be my whole set  $\theta$ .

(Refer Slide Time: 23:41)

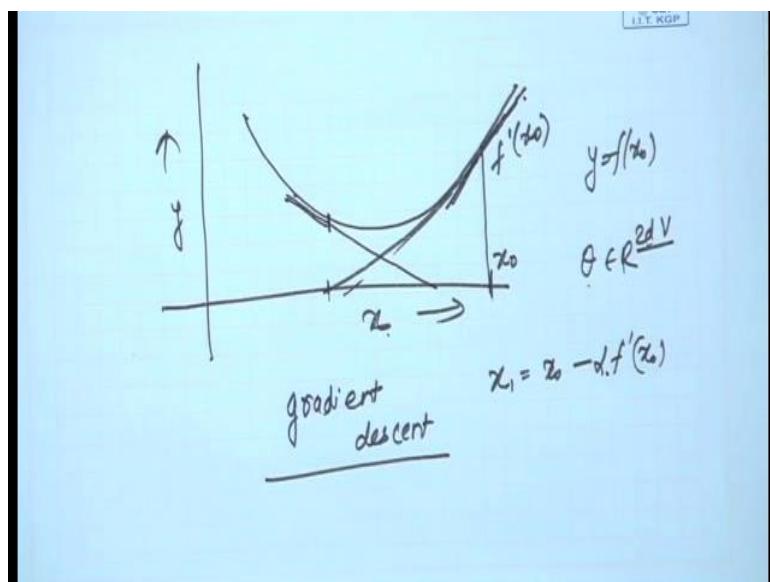
### Gradient Descent for Parameter Updates

$$\theta_j^{new} = \theta_j^{old} - \alpha \frac{\partial}{\partial \theta_j^{old}} L(\theta)$$

Pawan Goyal (IIT Kharagpur) Word Embeddings - Part II Week 7, Lecture 5 12 / 14

And what is the simple gradient descent formula. So this is like you take the derivative with respect to that particular parameter. So theta j new is theta j old minus alpha is my learning rate and you take a partial derivative of j theta with respect to the parameter when putting the old values.

(Refer Slide Time: 24:20)



Now, I am supposing that you know the idea gradient descent that is used to optimize yours minimize a particular function. So if I want to give this idea very briefly. So this is like suppose you have a simple function like this. And I want to find out for what parameter theta, of what parameter x will suppose, this is a function over x and this is my y. For what value of x this is minimized. So what I will do? I will start with N e x. So suppose I am starting with this x 0. I will find out the value of function. So I go to the function. So why is suppose f x 0.

Now I want to know what is the value of x where there is minimized. So what I will do? I will take the derivative of the function at this point f prime at x 0 and now, if I have to minimize if this direction of my gradient I have to go in the opposite direction of my gradient. So my new value should be x naught minus f prime x naught. So I will go into opposite direction and I will use some learning rate with what rate you will go in this direction and this is simply the idea of gradient. You keep on doing that. So you go somewhere again you find our suppose you are going at this point. So now, your gradient will point you to this direction. So will try to go in this direction again gradient will point

you some direction and finally, you will converge. So this is a very simple idea of gradient descent, but I will suggest that you have a look at it that what is actually the gradient descent.

So this is simple function, but your function can be over multiple parameters. So like here I have my theta with in 2d V dimension. So I have 2d times capital V many parameters. My theta is a function my j theta is a function of all these parameters. So what I will do, I take a partial derivative with respect to each of these parameters and accordingly update those parameters and this is how it will look like. So we are not going in in doing the derivation because that would not be in the scope of this course, but if you are interested you can try to find out if I derive this form what update values do I get for different what vectors how do I update these vectors.

(Refer Slide Time: 26:48)

The slide has a blue header bar with the text "Two sets of vectors". The main content area is white. At the top left, there is a purple box containing the text "Best solution is to sum these up". Below this, a mathematical equation is shown:  $L_{final} = L + L'$ . To the right of the equation, there is a circular profile picture of a man with glasses and a grey shirt. Below the equation, there is a purple box with the text "A good tutorial to understand parameter learning:" followed by a link: <https://arxiv.org/pdf/1411.2738.pdf>. Below that, there is a green box with the text "An interactive Demo" followed by a link: <https://ronxin.github.io/wevi/>. At the bottom of the slide, there is a dark footer bar with the text "Pawan Goyal (IIT Kharagpur)", "Word Embeddings - Part II", "Week 7, Lecture 14", and a set of small navigation icons.

And now once I have done that, I have 2 different vectors V and V prime. And I can simply some these over. And that will give me the final embedding for each word. And if you want to understand more about how do we learn these parameters. So you might have a look at this paper and this gives the nice tutorial for parameter learning and also if you want to try out an interactive demo, so you can try out here.

(Refer Slide Time: 27:23)

*Glove*

$$J = \frac{1}{2} \sum_{ij} f(P_{ij}) (w_i \cdot \tilde{w}_j - \log P_{ij})^2$$

Combine the best of both worlds – count based methods as well as direct prediction methods

- Fast training
- Scalable to huge corpora
- Good performance even with small corpus, and small vectors

Code and vectors: <http://nlp.stanford.edu/projects/glove/>

Pawan Goyal (IIT Kharagpur) Word Embeddings - Part II Week 7, Lecture 5 14 / 14

2

$$J = 1/2 \sum_{ij} f(P_{ij}) (w_i \cdot \widehat{w}_j - \log P_{ij})$$

So what to work is one famous model and such 2 variation c V W and skip gram. And both are used. So in some tasks skip gram has shown to be better in some CBOW has shown to be better. And there are many other tricks that I used for learning these parameters that we have not covered. I have only tried to give you the intuition. So I hope you will at least have an idea that how these word vectors are learned from my data, by just predicting neighboring words from the context or context from the neighboring word. And what is the embedding? So, how these weight vectors for my embeddings.

So now there are some other models, so one other popular model is glove model. So what is a basic cognition of glove model? So in this skip gram I am trying to learn my all my embedding from scratch. So in glove model what they are saying. So from a corpus you can kind of count the co-occurrence. See know with that is what we did in the distribution semantics, we would be counting co-occurrence. Now what they are saying? So this gives a very good idea about the words. So which words are similar to what are the words? And suppose you find out what is the co-occurrence probability for any towards. Now try to make word vectors such that when you do a dot product between towards you get close to the actual co-occurrence that you are seeing in the corpus.

And by doing that you will get a low dimensional representation and also it will take care of the words that are coming very sparsely, where you do not have enough data in the corpus to predict the co-occurrence. And this is the simple objective function. So that is if you look at the objective function. So forget word  $f$ ,  $f$  is a few simple function that make sure that if certain words are certain co-occurrence, we are very popular you do not base your learning towards those. See you just clip those at certain point. So everything above that will be converted to this one.

So this is the main idea here of this glove vectors. So you are having in word definition for  $i$  and  $j$  that you want to learn. An idea is try to learn them that are such that they are resembling their whatever you get from the co-occurrence probability. So  $P_{i,j}$  is from the co-occurrence. You can use either  $P_{m,i}$  condition probability or many other things. So find out  $w_i, w_j$  such that they are close to this co-occurrence probability. And then we have these objective function and you optimize this objective function and try to learn your different weights. So you are trying to combine the best of both words. Using the count based methods and the direct prediction methods. And using both of these you are trying to come up with your word vectors. And these vectors have also become very popular because the training is much faster than then my skip gram model because now you are not going to each individual word in your and all the windows. You have already computer the counts now you are only looking at the pairs word pairs and one maximizing that. And even with the small corpus and small vectors they have shown very good performance.

So one good thing is that suppose you want to use this vector finding of your task, so either word vectors or glove vectors are freely available on different websites. So what do we already told the website in the previous lecture for glove, you can go to this Stanford website and there you can download code as well as the vectors. And these vectors you can use for many of your tasks, where you are trying to find out whether 2 words are similar, or you are trying to find out some analogy task  $a$  is to  $b$  then  $c$  is to what. And many other applications they have been used in. So I think this is what I had to say for this tropical distribution semantics, this is a very well growing research field and lot of new thinks happening.

So I hope whatever we have discussed will help you to also understand the papers in this field if you want to going in further depth, and also you can try out these ideas for many

of your applications. So in the next week what we will do. So we will start with a separate notion of semantics that is Leska semantics. So how to use lexicon to find semantics between words.

Thank you.

**Natural Language Processing**  
Prof. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute Technology, Kharagpur**

**Lecture - 37**  
**Lexical Semantics**

Hello everyone. Welcome to the week eight of this course. In the last week we had started discussions on semantics and we discussed a particular method for caption semantics that were distribution semantics; so how we can use the distribution patterns of words in a corpus to extract meanings out of that. And we said that how we can compare two words to find out if they are more similar than another pair of words.

So, in this week what we will be doing, we will be taking another approach of semantics that is by using the connections that we see among items in lexicon. And that is where we will also see how to use a very important resource that is word net for extracting meaning some between words. So, topic for this week is Lexical Semantics.

(Refer Slide Time: 01:06)

**Lexical Semantics**

**Definition**

**Lexical semantics** is concerned with the systematic meaning related connections among lexical items, and the internal meaning-related structure of individual lexical items.

To identify the semantics of lexical items, we need to focus on the notion of **lexeme**, an individual entry in the lexicon.

**What is a lexeme?**

**Lexeme** should be thought of as a pairing of a particular orthographic and phonological form with some sort of symbolic meaning representation.

- Orthographic form, and phonological form refer to the appropriate form part of a lexeme
- Sense refers to a lexeme's meaning counterpart.

So, how can we define lexical semantics? So, as I am saying what we are doing in this topic, we will see some entries in a lexicon and we will try to identify what is the relation between these generatives. So, we will have some particular set of relations and we will try to define this relation among these entities. So, in that way we can define this field. So, lexical semantics is concerned with the systematic meaning related connections among lexical items.

So, I will have a certain lexical items in my lexicon, and I will try to find out what are the connections between any two items and we will be also see what are the internal meaning related is structure of individual lexical items.

So, what you are seeing here? We are not focusing our attention to something called a lexical item. So, what is the lexical item? So, lexical item I there is another term for that this called lexeme. So, this is nothing, but an individual entry in my lexicon. So, here I am talking about connections among different lexical items and what is the single entity it will call a lexeme and we will mostly we talking about in terms of lexemes. Now so if we try to understand intuitively, what would be there in an entry in a lexeme entry, what are the things that I need to keep in an entry? So, because I am also talking about the meaning, if a word as multiple meanings it should not be called like a single lexeme it should be called as two different lexemes.

So, when I talk about lexeme, what all things I need to have? I need to have what is the form; so how do I spell that, how do I pronounce that and also some something that says what is the meaning of this particular entry. So, these are the two important components that make like a lexeme. So, I will have. So, I can think of lexeme as a pairing of some orthographic and phonological form, that is how do you write the lexeme, how do you pronounce that and with some sort of symbolic meaning representation. Something that might be definition of the lexeme or something that explains what this it is stands for.

So, when I talk about orthographic form and phonological form, they are the form part of the lexeme and then I have a science part of the lexeme that talks about the meaning of the lexeme. So, if we have seen in various dictionaries that is all the entries are also defined.

(Refer Slide Time: 03:39)

**Example**

**verge** <sup>1</sup> | vərj |  
noun  
an edge or border: *they came down to the verge of the lake.*  
• an extreme limit beyond which something specified will happen: *I was on the verge of tears.*  
• BrE a grass edging such as that by the side of a road or path.  
• Architecture an edge of tiles projecting over a gable.

verb [ no obj. ] **verge on**  
approach (something) closely; be close or similar to (something): *despair verging on the suicidal.*

ORIGIN late Middle English: via Old French from Latin *virga 'rod.'* The current verb sense dates from the late 18th cent.

**verge** <sup>2</sup> | vərj |  
noun  
a wand or rod carried before a bishop or dean as an emblem of office.

ORIGIN late Middle English: from Latin *virga 'rod.'*

**verge** <sup>3</sup> | vərj |  
verb [ no obj. ]  
incline in a certain direction or toward a particular state: *his style verged into the art nouveau school.*

ORIGIN early 17th cent. (in the sense 'descend (to the horizon)': from Latin *vergere 'to bend, incline.'*



Ptwan Goyal (IIT Kharagpur)      Lexical Semantics      Week 8, Lec 18

So, let me take one example. So, this is example from the dictionary that comes with mac. So, you are having suggested single word like verge, but it appears size 3 different lexemes. So, we are having verge one, verge two and verge three and do I see the pair the form part in the meaning part? So, form part is giving on top for each entry. So, I have the spelling, so that is the same for all 3, then I have the pronunciation also; then there are certain descriptions about this entry that that correspond to what is the meaning of this entry.

For example you might have the lexical creditory of that. So, it is a noun, it is a noun, it is a verb and then there is some meaning like an edge or border they come down to the verge of the lake. So, also given an example how it is used and some history of the words also provide it. So, when I talk about lexeme that is what I need. I need a form part that can be the spelling of the lexeme plus the pronunciation, and the meaning part. Once I have done that now we can try to establish relations between various lexemes in my lexicon or each lexical item in my lexicon.

So, now what are the various meaning related facts you can extract from the, with the lexemes and defined in my dictionary.

(Refer Slide Time: 05:04)

*Definitions from the American Heritage Dictionary (Morris, 1985)*

- **right** *adj.* located near the right hand esp. being on the right when facing the same direction as the observer
- **left** *adj.* located near to this side of the body than the right
- **red** *n.* the color of blood or a ruby
- **blood** *n.* the red liquid that circulates in the heart, arteries and veins of animals

The entries are description of lexemes in terms of other lexemes

Definitions make it clear that *right* and *left* are similar kind of lexemes that stand in some kind of alternation, or opposition, to one another

We can glean that *red* is a color, it can be applied to both *blood* and *rubies*, and that *blood* is a liquid.

Pawan Goyal (IIT Kharagpur)      Lexical Semantics      Week 8, Lecture 1      4 / 18

So, let us take some examples from this heritage American heritage dictionary. So, is 1985 version and what we are seeing here; we are seeing four different entities here right, left, red and blood they are 4 different lexemes in my in this dictionary and some meaning of these entities are also provided. So, for example, right is defined as located near the right hand especially being on the right when facing the same direction as the observer; and left is defined as located near to this side of the body then the right. So, this is some definitions provided to these lexemes.

Now, what is something that you can immediately point that you can immediately see from these definitions? So, one thing that you see is that the word right has been defined terms of right; this is some sort of circular definitions, if you say in that way and also lexemes are defined in terms of other lexemes. So, like here red the color of blood or a ruby and blood is defined the red liquid. So, blood is defined in terms of red and red is defined in terms of blood. So, we see that there is some sort of circularity involved here in the way we have defined in the lexicon.

But even if that is the case you can still extract some sort of meanings for example, if you see the definitions of right and left, you can easily find out that they are somewhat related, their meanings are very much related and you might also say that they might be in some sort of ultravibration with respect to each other.

So, what do we see here? So, the entries are description of lexemes in terms of other lexemes and from the definitions I can see that right and left are some similar sort of lexemes and they are in some sort of alternation or opposition. Also from these definitions we can see that red is some sort of color and it can be apply to blood as well as ruby and blood is some liquid. So, you might extract some other facts about the lexemes also from these entries.

Now, in general; so when I am talking about identifying what is the connections of different between different entries, what are the various formal connections that have been defined, and what are the relations that we can study?

(Refer Slide Time: 07:41)

The slide has a blue header bar with the text "Relations between word meanings". The main content area contains a bulleted list of eight semantic relations:

- Homonymy
- Polysemy
- Synonymy
- Antonymy
- Hypernymy
- Hyponymy
- Meronymy

At the bottom of the slide, there is a footer bar with the text "Ptwan Goyal (IIT Kharagpur)" on the left, "Lexical Semantics" in the center, and "Week 8, Lecture 1" and "5/18" on the right. There are also small navigation icons for a presentation slide.

So, let us have a look at some of the relations. So, these are some very important relations that are established between word meanings and forms. So, like Homonymys Polysemy, Synonymy, Antonymy, Hypernymy, Hyponymy and Meronymy and you might have heard of some of the list terms like Polysemy which very very popular and similarly synonymy and Antonymy so these we might already have heard. So, let us try to formally define what are these relations and when do I say that to and it is my lexicon are connected by a particulation from one of these

(Refer Slide Time: 08:16)

**Homonymy**

**Definition**

**Homonymy** is defined as a relation that holds between words that have the same form with unrelated meanings.

**Examples**

- Bat (wooden stick-like thing) vs Bat (flying mammal thing)
- Bank (financial institution) vs Bank (riverside)

**homophones and homographs**

**homophones** are the words with the same pronunciation but different spellings.

- write vs right
- piece vs peace

**homographs** are the lexemes with the same orthographic form but different meaning. Ex: bass

Pawan Goyal (IIT Kharagpur)      Lexical Semantics      Week 8, Lecture 1      6 / 18

So, how do I define homonymy? So, homonymy is defined as a relation that holds between words that have the same form, but unrelated meanings. So, what is that mean? So, we have defined an individual entry in my lexicon as having a form part and a meaning part. So, now, if two entities have the same form part, but the meaning part is different then they will call them homonyms. Now can you think of an example of two such or a single word that has the same form, but two different meanings. So, simple example might be like the word bank. Bank and before the financial institution bank or it can be some river bank. Similarly bat it can be for the cricket bat or it can be the bat as a mammal. So, what you are seeing here? They are having the same form, but different meanings.

So, these are two examples that we talked about. Now so there is one important thing that you should remember here. So, we are seeing that they have the same form, but different meanings, but we have defined form to be both orthography form and phonological form. So, it might happen that in certain cases, the orthography form is same and meaning is different and in some cases the phonological form is same and meaning is different; they are called homonyms, but there are specific terms for both of these and these are called homographs, if there is the same orthography and homophones if there is the same phonological form.

Let us see some examples, so homophones are the words that have the same pronunciation, but different spellings, and different meanings also. So, examples here are write and right; there is the same homological form, they are written differently and their meanings are

different. Same with piece and peace and homographs would be the lexemes and that the same orthographic form, then they would have may be the different pronunciation and different meaning all to gather. So for example, the two different usage of bass, one in the sense of (Refer Time: 10:28) in the sense of a guitar or music.

(Refer Slide Time: 10:34)

**Problems for NLP applications**

- Text-to-Speech**  
Same orthographic form but different phonological form
- Information Retrieval**  
Different meaning but same orthographic form
- Speech Recognition**  
to, two, too  
Perfect homonyms are also problematic

Pawan Goyal (IIT Kharagpur)      Lexical Semantics      Week 8, Lecture 1      7 / 18

Now, now why do we need to worry about this homonymys? So, we will see, what are the different NLP applications, where they create a problem? So, let us take this example of text to speech. So, do you think these homon, homophones or homographs would create a problem for text to speech? So, what do we do in text to speech? In text to speech you would have a text contain written, it can be a sentence where there are multiple words together and you need to speck that out. So, why the homophones or homographs can create a problem for that now think of the same word that as the particular spelling, but it is pronounce into different ways into different meanings. So, unless you know what is the particular sense it is being implied you cannot pronounce it properly.

So, if the of orthographic form is same, but the phonological forms are different that creates a problem for text to speech. Now it can also create the problem for information retrieval; why the problem for information retrieval? So, suppose you are looking for the term bat and by bat you implied the mammal, but sup the suppose the systems gives you cricket bat as the desired or as the page edge that that you will look for, and immediately it will not match your information need.

So, what would happen if the same word has multiple meanings, if you are searching for that word, because you are only specifying the orthographic form, it may not be clear to the search engine what is the meaning that you are looking for. So, these also create a problem for the search engines.

And it might be a problem for the speech in (Refer Time: 12:26) also for example, the perfect homonyms or homophones, where you have the same phonological form, but different ways of writing that. So, suppose you are writing to suppose you are speaking to. So, it may not be clear, which of the 3 words should be the correct interpretation, whether it is to or too or two. So, these are also problematic for speech reorganization. So, these are some problems where they create problems. Now suppose the systems have to deal with that.

So, how do they find out when I am talking about to is it to or is it two or double o and for that they have to use various information from the context in which that is spoken or something about the speaking and so on and some of the problems of this kind we will also deal in this week in the topic of word sense disambiguation, that if a word has multiple meanings how do I use the context to identify, what is the particular meaning is being used here.

(Refer Slide Time: 13:44)

**Polysemy**

**Multiple related meanings within a single lexeme.**

- The **bank** was constructed in 1875 out of local red brick.
- I withdrew the money from the **bank**.

**Are those the same sense?**

- Sense 1: "The building belonging to a financial institution"
- Sense 2: "A financial institution"

**Another example**

- Heavy snow caused the roof of the **school** to collapse.
- The **school** hired more teachers this year than ever before.

Pawan Goyal (IIT Kharagpur)      Lexical Semantics      Week 8, Lecture 1 / 8

So, now coming to the second relation that is Polysemy; So, Polysemy and homonymy might be confused with certain times. So, Polysemy is also the same sort of relation that is a word with multiple meanings, but the same form, but the difference here is these meanings are very

related and not unrelated tight we had in the case of homonyms. So, in case of homonyms, 2 words have same form, but very different meanings for Polysemy they have the same form and slightly related meanings not very very similar and not very very different also.

So, what is the example? Let us focus on the Word bank here in the two sentences in the first sentence we see, the bank was constructed in 1875 out of local red brick. And second one is the sentence I withdrew the money from the bank. So, as such if you look at these sentences you may feel that. So, there the same Word bank is being used in for the same sense, but if you look closely you find there is some slightly some slight difference between the meanings of the Word bank. So, what is the difference here? In the first sentence the bank is used as some sort of building that belongs to a financial institution.

On the other end second sentence it belongs to the financial institution itself and that sort of slight gradation in meaning is there in many such words. For example, let us look at these two sentences, heavy snow caused the roof of the school to collapse and the school hired more teachers this year than ever before. So, again the same word school is being used as they look the same sense, but again if you look closely, this refers to the building of a school and this refer to the institution a school as such or their administration.

(Refer Slide Time: 15:55)

*Polysemy: multiple related meanings*

*Often, the relationships are systematic*

E.g., building vs. organization  
school, university, hospital, church, supermarket

*More examples:*

- Author (Jane Austen wrote <sup>the</sup> Emma) ↔ Works of Author (I really love Jane Austen)
- Animal (The chicken was domesticated in Asia) ↔ Meat (The chicken was overcooked)
- Tree (Plums have beautiful blossoms) ↔ Fruit (I ate a preserved plum yesterday)

Navigation icons: back, forward, search, etc.

Page footer: Pawan Goyal (IIT Kharagpur), Lexical Semantics, Week 8, Lecture 1, 9 / 18

So, and these relations are called systematic. So, for example, if you look at these words school, university, hospital, church, supermarket all of these can be seen as the building as such or the institution and that is why the two meanings would be slightly related, they are

not very different and not exactly the same. There are some other examples like you can have the make the connection between the author and the works of author. So, if I say Jane Austen wrote Emma some talking about the author, on the other hand if I say I really love Jane Austen. So, what do your meaning is that? I really love the works of the author. So, author in works of author again appearing some sort of slightly different meaning side that is that is observed by this term Polysemy.

Similarly animal versus meat; so, when you (Refer Time: 16:52) chicken it can be animal or meat. Similarly sometimes it can be tree versus fruit and this kind of relation like if I say plums. So, in these sentence plums have beautiful blossoms, plums mainly refer to the tree, but here I ate a preserved plum, plum would be a fruit. So, again there are very very slight distinctions between the meanings here.

(Refer Slide Time: 17:17)

**Polysemy: multiple related meanings**

**Zeugma test**

- Which of these flights *serve* breakfast?
- Does Midwest Express *serve* Philadelphia?

\* Does Midwest Express *serve* breakfast and San Jose?

Combine two separate uses of a lexeme into a single example using conjunction

Since it sounds weird, we say that these are two different senses of *serve*.

Pawan Goyal (IIT Kharagpur)      Lexical Semantics      Week 8, Lecture 18

Now, in linguistics, suppose I want to distinguish identify if the same word is being used in different sentences, it is been used in slightly different meanings or is the word Polysemy somehow. So, what is one particular linguistic test that we can do and this test is called zeugma test and what is the idea? Suppose the same word has been used in two different sentences; now try to compose a single sentence where the word has been used only once and the two different usages have been captured by some sort of consumption, and see if this sentence is making sense already is looking weird. If the sentence is making sense so

probably this word as the same sense in both the sentences, but if the sentences looks weird then you (Refer Time: 18:11) there are Polysemys.

So, here is an example. So, I have the same words serve being used in two different sentences. So, which of these flights serve breakfast and does Midwest express serve Philadelphia? Now I want to find out if there is some Polysemy involved in the usage of the word self here. So, what I would do? I would combine these 2 usages together in a single sentence. So, what can be one sentence we can compose out of this? So, you can compose like does Midwest express serve, breakfast in Philadelphia and immediately we are (Refer Time: 18:48) serving breakfast in Philadelphia putting them together in the same sentence, looks weird. You can put them in to a different sentence, but putting them in same sentence looks weird then we can say that. So, these two usage of the words are Polysemys. So, here does Midwest mist west express serve breakfast and San Jose? If you put them together, it sounds weird so we say that they are two different senses of the word serve.

So, that is what we have covered Homonymy, Polysemy remember they are very very it may not be easy to distribution among the two; in Polysemy they are quite related, in Homonymy they are very very different. In general if you might also, you might in general you might say that word is (Refer Time: 19:38) you note that the word has multiples senses. Now the next relation that we will see is synonymy, what is synonymy; two words that have very very similar meanings but different orthographic and phonological form.

(Refer Slide Time: 20:00)

**Synonymy**

*Words that have the same meaning in some or all contexts.*

- filbert / hazelnut
- couch / sofa
- big / large
- automobile / car
- vomit / throw up
- water /  $H_2O$

Two lexemes are synonyms if they can be successfully substituted for each other in all situations.

Pawan Goyal (IIT Kharagpur)      Lexical Semantics      Week 8, Lecture 1      11/18

So, examples are like couch and sofa; they have same meaning different words big and large, automobile car, water H<sub>2</sub>O. So, now, what is important here is these two words are similar meanings, but probably they are not substitutable in all the contexts. For example, in scientific context you might want to write H<sub>2</sub>O, but not water. But when you are all talking in general terms you would use the term water crossing you to give you water, you will not ask give you H<sub>2</sub>O. So, all though the meanings might be very very singular, depending on the context you use one word or the other.

So, as such we can call two lexemes are synonyms, if they can be successfully substituted for each other in all situations, but this not actually the case we cannot; in general you cannot substitute them in all the places you can probably do them most to the places, but not (Refer Time: 20:58).

(Refer Slide Time: 21:01)

**Synonymy: A relation between senses**

Consider the words *big* and *large*.

**Are they synonyms?**

- How **big** is that plane?
- Would I be flying on a **large** or small plane?

**How about here?**

- Miss Nelson, for instance, became a kind of **big** sister to Benjamin.
- \*Miss Nelson, for instance, became a kind of **large** sister to Benjamin.

**Why?**

- *big* has a sense that means being older, or grown up
- *large* lacks this sense

Pawan Goyal (IIT Kharagpur)      Lexical Semantics      Week 8, Lecture 1      12 / 18

So, now, let us take an example. So, I take two words big and large and let us see these sentences; how big is that plane? I can also write how large is that plane. So, would I be flying on a large or small plane? So, with word plane I can use the word either big or large and both are substitutable here.

So, in that sense they look like perfect synonymys, but can I always replace big for large or large for big in all the contexts. Let us see this example; suppose I have the sentence Miss Nelson, for instance became a kind of big sister to Benjamin and suppose I replace big by large here, and the sentence comes out to be Miss Nelson for instance became a kind of large

sister to Benjamin and immediately you can see that this usage of word large, immediately it starts looking weird and why is that the case? So that means, we cannot substitute large for big, big here and why is that?

So, because you would see that big has a very very specific sense that is being grown up or elder and that sense never applies to the word large. So, big has a sense of being older or grown up, but large as such lacks this sense, it does not have this sense.

(Refer Slide Time: 22:29)

**Synonyms**

*Shades of meaning*

- What is the cheapest first class *fare*?
- \*What is the cheapest first class *price*?

*Collocational constraints*

- We frustate 'em and frustate 'em, and pretty soon they make a *big* mistake.
- \*We frustate 'em and frustate 'em, and pretty soon they make a *large* mistake.

Pawan Goyal (IIT Kharagpur)      Lexical Semantics      Week 8, Lecture 1      13 / 18

Sometimes they are very shades of meaning and some sort some words of being preferred to use them other words, like when you are talking about what is the cheapest first class fare? We will talk about you will prefer to say fare then price. Sometimes there are Collocational constraints. So, with some particular words only one word goes not the other one, even if they have very very similar meanings.

For example before mistake; would you like big mistake or large mistake, generally big mistake is preferred. So, here this is the sentence. So, we frustrate em and frustate e m and pretty soon they make a big mistake and we would not substitute large here. So, large mistake is not Collocational, but big mistake is Collocational. So, because of language usage also some words are preferred to be used then others. So, they cannot be substituted at all places probably in some places.

(Refer Slide Time: 23:32)

**Antonyms**

- Senses that are opposites with respect to one feature of their meaning
- Otherwise, they are similar!
  - ▶ dark / light
  - ▶ short / long
  - ▶ hot / cold
  - ▶ up / down
  - ▶ in / out

*More formally: antonyms can*

- define a binary opposition or at opposite ends of a scale (*long/short, fast/slow*)
- Be **reversives**: *rise/fall*

Pawan Goyal (IIT Kharagpur)      Lexical Semantics      Week & Lexis

Now, coming to the next relation that is antonyms; so here I say that; so, what is the general notational the general definition of antonyms two words if they are entirely opposite to each other they are opposite to each other; but this is something interesting about antonyms. So, two words antonyms if they are completely; so we said that they are having very very opposite meanings, but in general if you look closely, they will have very very related meanings, but they are opposites only in word particular aspects. So, let us take one example, see like hot and cold. So, do you call them are they related or are they very very different. So, as such they are related, they both talk about temperature yes, but they are at the opposite ends of the history extreme. So, we will say it is cold one extreme and hot is the other extreme, but as such both talk about temperature.

And suppose you try to use the distribution semantic method that we tried in the last week, to capture if two words are antonyms. It might become very very difficult, because we would occur in very very similar contexts yes. So, hot and cold occur with words like weather yes temperature and so on. So, what we are saying in antonyms, they are having very very similar meaning, but they are opposite only in one aspect of the meaning. So, like dark and light, short and long, hot and cold, up and down, in and out; they are opposite only in one aspect otherwise they are similar.

So, more formally we can say that antonyms can define some sort of binary opposition or at opposite ends of a scale. So, like long and short, hot and cold, fast and slow, and be reversives it is like rise and fall.

(Refer Slide Time: 25:49)

**Hyponymy**

One sense is a hyponym of another if the first sense is more specific, denoting a subclass of the other

- car is a hyponym of vehicle
- dog is a hyponym of animal
- mango is a hyponym of fruit

**Hypernymy**

Conversely

- vehicle is a hypernym/superordinate of car
- animal is a hypernym of dog
- fruit is a hypernym of mango

Now, coming to another important relation that is Hyponyms and Hypernyms; now what are Hyponyms and Hypernyms? So, they are relations between a subclass and a super class. So, formally we say one sense is a hyponym of another, if the first sense is more specific and denotes a subclass of the other. So, example here is like car, car is a subclass of vehicle, so we can say that car is a hyponym of vehicle. Similarly dog is a class of animal, so we say dog is a hyponym of animal and mango is a hyponym of fruit, because mango is a type of fruit.

Now, just the other way round the relation would be called hyponymy. So, I say mango is a type of fruit, so fruit is a super class. So, mango is a hyponym of fruit and fruit is a hyponym of mango. So, conversely we can define the hypernymy relation also. So, vehicle is a hypernym of car, and animal is a hypernym of dog, and fruit is a hypernym of mango and these are of some of the relations that we will see very closely what is. So, how do we use hypernymy relation for extracting various meaning related facts among entities?

(Refer Slide Time: 27:08)

*Hyponymy more formally*

*Entailment*  
Sense A is a hyponym of sense B if being an A entails being a B.  
Ex: dog, animal

*Transitivity*  
A hypo B and B hypo C entails A hypo C

Ptwan Goyal (IIT Kharagpur)      Lexical Semantics      Week & Lecture 18

So, with hyponyms we can also define some formal criteria. So, for example, entailment; if I say that sense is A hyponymy of sense B; that means, being an A entails being a B. For example, as a dog is a hyponymy of animal dog is subclass, animal is super class. So, being a dog entails being an animal; as it is some entities dog, it is an animal. So, it will have all the characteristics of animal also impose on that and, but you cannot say the other way round, that is some if someone something is a dog animal, it is also called dog this is not valid.

So, if a is A hyponymy of B. I will say being an A entails being a B. You can also define the transitivity relation that is if A is hyponymy of B, B is a hyponymy of C then A is also hyponymy of C; that is also very very trivial condition on hyponyms; so entailment in transitivity.

(Refer Slide Time: 28:16)

**Meronyms and holonyms**

**Definition**

**Meronymy:** an asymmetric, transitive relation between senses.  
X is a **meronym** of Y if it denotes a part of Y.  
The inverse relation is **holonymy**.

meronym	holonym
porch	house
wheel	car
leg	chair
nose	face

Pawan Goyal (IIT Kharagpur)      Lexical Semantics      Week 8, Lecture 1      17 / 18

Now, we can also define other relations like Meronymys and Holonyms. So, Meronymy is some sort of asymmetric, but transitive relation between senses what is that? So, I will say X is a meronym of Y, if X denotes a part of Y. So, it is not symmetric; so example is. So, here porch is a part of house or wheel is a part of car. So, to define the part of relation I say wheel is a meronym of car, leg is a meronym of chair, nose is a meronym of face, and the other where round relation is called Holonym that is face is a holonym of nose, car is a holonym of wheel and so on.

So now; so we saw what is the different relation that we can defined among lexical entities. So now, what are the different lexical sources? Where these relations are nicely captured? So, where you can find out what entries are related by the relation of hyponymy, hypernymy and meronymy and use for different tasks or application that you want to bit. And that is what we will see in the next lecture, then we will have a discussions on what is the structure of word net, and how are all these relations defined in word net and how can you can use that to capture various semantics like whether these two words are related or not and to word degree. That will be the topic for the next lecture.

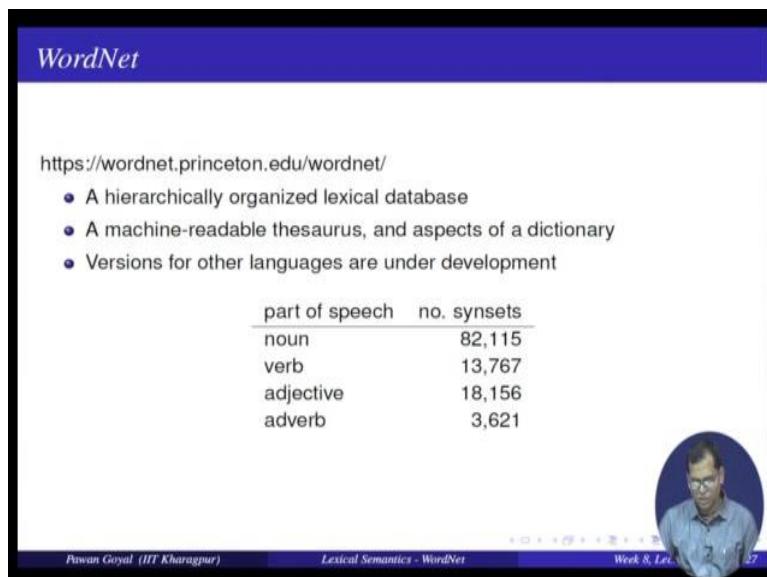
Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 38**  
**Lexical Semantics – Wordnet**

Yes, welcome back to the second lecture of this week. So, in the last lecture, we started with lexical semantics and we defined various relations between lexical entities, and we talked about polysemy, hyponym, hypernym, meronymy and so on. And in this lecture, we will see a particular resource wordnet and how it captures all these relations.

(Refer Slide Time: 00:48)



The screenshot shows the homepage of the WordNet website. The title 'WordNet' is at the top. Below it is a URL: <https://wordnet.princeton.edu/wordnet/>. A bulleted list describes WordNet as a hierarchically organized lexical database, a machine-readable thesaurus, and aspects of a dictionary, with versions for other languages under development. Below this is a table showing the number of synsets for different parts of speech:

part of speech	no. synsets
noun	82,115
verb	13,767
adjective	18,156
adverb	3,621

At the bottom, there is a circular profile picture of Prof. Pawan Goyal, along with the text 'Pawan Goyal (IIT Kharagpur)', 'Lexical Semantics - WordNet', 'Week 8, Lec 38', and the number '27'.

So, wordnet, you can get a lot of information of wordnet on this website. So, it is an effort started at Princeton; and there are lot of versions that have come up. And you can find out the latest version of wordnet and also download that and you can use that also, once you download you can use that in your command terminal also. So, what is wordnet? Wordnet as such is a hierarchically organized lexical database and it is completely machine readable.

So, you can use that in different applications, you can call wordnet get the recent information from there. So, as such on this website, you will find the wordnet for English, but there are other so there are many, many other versions for other languages also built. So, there are lot of wordnets available for European languages, and a lot of effort has happened in the last decade for building wordnets for Indian languages and that you can also download many of

those versions are also available for download for free. So, for that you can look at indo wordnet website. So, there you will also find out what sort of synsets or concepts in one language are relate to some other concepts in other language. So, this information would be available in this indo wordnet and euro wordnet websites.

So, now we will focus only on the English wordnet part in this lecture, but the methods would be easily applicable to other wordnets that are having a very similar structure. So, if we talk about English wordnet, so there are mainly four different part of speech words that are present there. And what we are seeing here; how many different synsets are there for each part of speech. So, there are roughly 18,000 synsets for nouns, 13,000 plus synsets for verb, and 18,000 plus synsets for adjective, and 3,000 plus for adverbs now. So, one important thing is that we when we talk about wordnet we talk in terms of synset. Now, what is the idea of a synset in wordnet. So, in synset what will happen, so various different words that have similar meanings will be stored, but a single word might have multiple meanings also. So, how are both of these things taken care together?

(Refer Slide Time: 03:18)

**Synsets in WordNet**

- A **synset** is a set of synonyms representing a sense
- Example: chump as a noun to mean 'a person who is gullible and easy to take advantage of'  
{chump<sup>1</sup>, fool<sup>2</sup>, gull<sup>1</sup>, mark<sup>9</sup>, patsy<sup>1</sup>, fall guy<sup>1</sup>, sucker<sup>1</sup>, soft touch<sup>1</sup>, mug<sup>2</sup>}
- Each of these senses share this same gloss.
- For WordNet, the meaning of this sense of chump is this list.

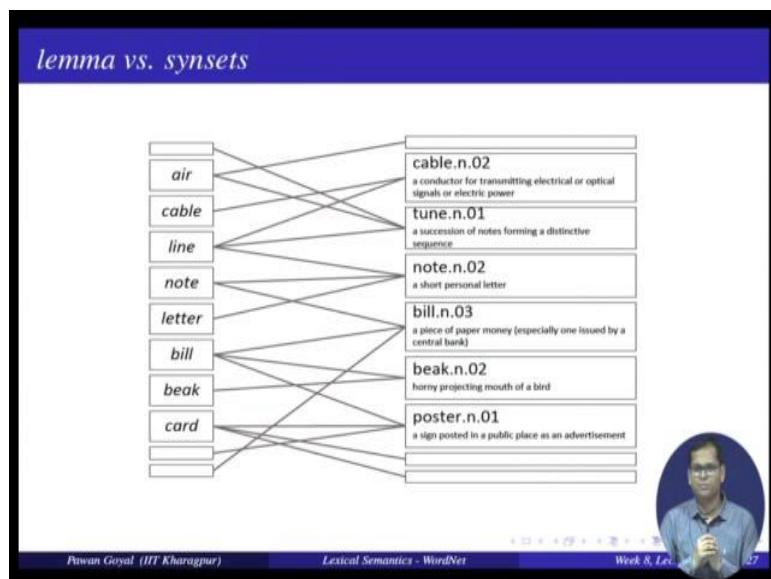
Pawan Goyal (IIT Kharagpur)      Lexical Semantics - WordNet      Week 8, Lec. 17

So, let us take an example. So, example is particular synset corresponding to chump that is a noun that means a person who is a gullible and easy to take advantage of. And suppose chump the first synset has this particular meaning. So, word has have might have multiple synsets. So, chump first synsets is this meaning, but there might be other words also that have the sense. So, this word, its 9 sense denotes this particular meaning. So, I have to say that the

word mark its 9th sense is this meaning; the word fool, its second sense is this meaning; the word gull the first sense is this meaning, and similarly fall guy and so on.

So, what I will do here? Each word might be represented by multiple synsets. So, a word might have sense 1, sense 2, sense 3 and so on. And what I will do, if suppose the third sense of word 1 and second sense of word 2 shared the same meaning I will put them in a single synset and that is the idea of synset, different words that share a same meaning. And by words here I will mean the lexical or a particular sense. And for wordnet this list might denote what is the meaning of the word chump or you will also find some gloss information of the wordnet.

(Refer Slide Time: 04:50)



So, now this also this figures some sort of explains what is the relation between the word form or the lemma and the synsets. So, what you are seeing here? So, there is many to many relation. So, that is for example, the word note, it can go to note n 2 and bill n 3. So, denotes two different synsets in one sense, it goes to note synsets another sense it goes to bill synset. But this bill synset will contain note and also contains bill and some other words. So, what you are seeing here? Many different lemmas that share the same sense can come together in a single synset; and the same lemma for different meanings can go to different synsets, and that is captured by giving some unique identifies to different two different synsets of the same lemma. So, I will start talking about now line 1, line 2, line 3 instead of just saying line; and each of these three will be in different synsets.

(Refer Slide Time: 06:02)

### All relations in WordNet

searchtype is at least one of the following:

- ants(n|v|a|r) Antonyms
- hypon(n|v) Hypernyms
- hypo(n|v), -tree(n|v) Hyponyms & Hyponym Tree
- entav Verb Entailment
- syns(n|v|a|r) Synonyms (ordered by estimated frequency)
- smenn Member of Holonyms
- ssubn Substance of Holonyms
- sprtn Part of Holonyms
- membn Has Member Meronyms
- subsn Has Substance Meronyms
- partn Has Part Meronyms
- meron All Meronyms
- holon All Holonyms
- causv Cause to
- pert(a|r) Pertaining
- attr(n|a) Attributes
- deri(n|v) Derived Forms
- domn(n|v|a|r) Domain
- dome(n|v|a|r) Domain Terms
- famln(n|v|a|r) Familiarity & Polysemy Count
- framv Verb Frames
- coor(n|v) Coordinate Terms (sisters)
- simsv Synonyms (grouped by similarity of meaning)
- hmern Hierarchical Meronyms
- hholn Hierarchical Holonyms
- grep(n|v|a|r) List of Compound Words
- over Overview of Senses
- 



Pawan Goyal (IIT Kharagpur) Lexical Semantics - WordNet Week 8, Lecture 2 / 27

So, what are all the possible relations that you see in wordnet here is some example. So, in wordnet, you have relations like antonyms, hypernyms, hyponyms, entailment relations, synonyms and so on, many of these we have already seen in the previous lecture.

(Refer Slide Time: 06:26)

### Wordnet noun and verb relations

Relation	Also called	Definition	Example
Hypernym	Superordinate	From concepts to superordinates	<i>breakfast<sup>1</sup> → meal<sup>1</sup></i>
Hyponym	Subordinate	From concepts to subtypes	<i>meal<sup>1</sup> → lunch<sup>1</sup></i>
Member Meronym	Has-Member	From groups to their members	<i>faculty<sup>2</sup> → professor<sup>1</sup></i>
Has-Instance		From concepts to instances of the concept	<i>composer<sup>1</sup> → Bach<sup>1</sup></i>
Instance		From instances to their concepts	<i>Austen<sup>1</sup> → author<sup>1</sup></i>
Member Holonym	Member-Of	From members to their groups	<i>copilot<sup>1</sup> → crew<sup>1</sup></i>
Part Meronym	Has-Part	From wholes to parts	<i>table<sup>2</sup> → leg<sup>3</sup></i>
Part Holonym	Part-Of	From parts to wholes	<i>course<sup>1</sup> → meal<sup>1</sup></i>
Antonym		Opposites	<i>leader<sup>1</sup> → follower<sup>1</sup></i>

Relation	Definition	Example
Hypernym	From events to superordinate events	<i>fly<sup>9</sup> → travel<sup>3</sup></i>
Troponym	From a verb (event) to a specific manner elaboration of that verb	<i>walk<sup>1</sup> → stroll<sup>1</sup></i>
Entails	From verbs (events) to the verbs (events) they entail	<i>snore<sup>1</sup> → sleep<sup>1</sup></i>
Antonym	Opposites	<i>increase<sup>1</sup> ↔ decrease<sup>1</sup></i>

Pawan Goyal (IIT Kharagpur) Lexical Semantics - WordNet Week 8, Lecture 2 / 27

Let us see some example types of these relations. So, like what are the relations between various nouns. So, the relation hyponym that we discussed, so hyponym is relation between a concept and its super concept. So, like breakfast is a kind of meal. So, as was a meal is a hyponym of breakfast. So, this relation will be there in the wordnet where the breakfast 1

synset or the corresponding sense it is connected to meal one by the relation of breakfast is a hyponym of meal and meal is a hyponym of breakfast. Similarly, the converse relation of hyponym meal and lunch, member, so professor is a member of faculty; has- instance, is composer and instance is Bach; Austen is a instance of author; member holonym like capital is a member of crew; part meronym from whole two parts. So, table to leg. Part holonym the other way round from parts to wholes, course to meal, yes, and antonym that is a opposite leader and follower.

So, what would happen? All these relations are defined in wordnet, you know what particular relation means and then you can find out for a particular relation, what are the different pairs or given pair in the wordnet is there any relation that is defined. Similarly, there are relations between verb elements also like hyponymy, fly and travel. So, fly is a kind of travel. So, travel would be the hyponym for fly. Troponymy from a verb to a specific manner for that verb like, walk and stroll, so stroll is a typical manner of or a particular manner of walking. There can be entailment relation also from verbs to the words that they entail like snoring and sleeping. So, I will say snore and tell sleeping. So, this is also relation that is captured in wordnet, and then there can be antonyms for the opposites, so increase and decrease, they are antonyms for each other.

(Refer Slide Time: 08:51)

The screenshot shows the 'WordNet Hierarchies' interface. At the top, it says 'Synonyms/Hypernyms (Ordered by Estimated Frequency) of noun mouse'. Below this, it indicates '4 senses of mouse'. The first sense, 'Sense 1', corresponds to the common house mouse. Its hierarchy is as follows:

- => rodent, gnawer
- => placental, placental mammal, eutherian, eutherian mammal
- => mammal, mammalian
- => vertebrate, craniate
- => chordate
- => animal, animate being, beast, brute, creature, fauna
- => organism, being
- => living thing, animate thing
- => whole, unit
- => object, physical object
- => physical entity
- => entity

The second sense, 'Sense 4', corresponds to a computer mouse. Its hierarchy is as follows:

- => mouse, computer mouse
- => electronic device
- => device
- => instrumentality, instrumentation
- => artifact, artefact
- => whole, unit
- => object, physical object
- => physical entity
- => entity

At the bottom of the slide, there are navigation icons and text: 'Pawan Goyal (IIT Kharagpur)', 'Lexical Semantics - WordNet', 'Week 8, Lecture 2', and '7 / 27'.

Further, in wordnet, we can also capture what is the complete hierarchy of a given concept. And what do I mean by hierarchy that is starting from the root word and wordnet how do you

go down to that particular word. What are the different concepts that you need to pass through, and that you can find for all the different synsets for a given word, so that is why we said that wordnet is very, very hierarchically organized database. So, you can find out or you can locate each concept or a subset each synset in that complete hierarchical tree.

So, for example, if I see the word mouse, so I will find it has four sense. So, for sense one – mouse, you can find out the complete hierarchy. So, mouse is a kind of rodent, kind of placental, kind of mammal, vertebrate and so on up to animal, living thing, object physical entity, and entity you can go to the root of the wordnet entity, it starting from this particular sense of mouse. Here is another sense of mouse that is a computer mouse. Again you can keep on going up the hierarchy electronic device, device, artefact, object, physical object, physical entity, entity. And you can see where do they depart. So, mouse comes under whole then an artefact, and this mouse comes under living thing that is where they depart in that wordnet hierarchical tree. And this we can do for any sense any word in the wordnet, you can find out the complete hierarchy.

(Refer Slide Time: 10:32)

The slide has a purple header bar with the title "Word Similarity". The main content area contains the following bulleted list:

- Synonymy is a binary relation
  - Two words are either synonymous or not
- We want a looser metric
  - Word similarity or
  - Word distance
- Two words are more similar
  - If they share more features of meaning
- Actually these are really relations between **senses**:
  - Instead of saying "bank is like fund"
  - We say
    - ★ Bank<sup>1</sup> is similar to fund<sup>3</sup>
    - ★ Bank<sup>2</sup> is similar to slope<sup>5</sup>
- We will compute similarity over both words and senses

At the bottom of the slide, there is a footer bar with the following text: "Pawan Goyal (IIT Kharagpur)" on the left, "Lexical Semantics - WordNet" in the center, and "Week 8, Lecture 2" and "8 / 27" on the right.

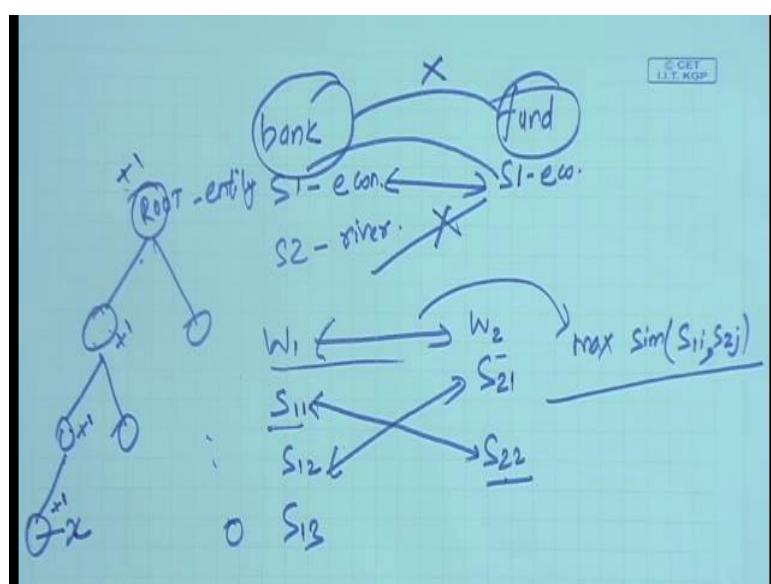
So, as such wordnet will always store which two concepts are related by word relations. And suppose car and automobile occur in the same synset, if you query in a wordnet, there is a very easy way you can query in wordnet, and you can find out if they are part of the same synsets then you can say ok they are similar. But suppose two words are not part of any sense any particular synset in wordnet, and I want to give it a number like to what degree are they

similar, similar to what we did in the case of distribution semantics, we found out how much two words are similar to each other, by seeing how much their patterns are similar. So, can I do something in the case of wordnet that is given two concepts or two words how similar they are even if they are not part of the same synsets.

So, by using synonymy I can only say whether that the two words are synonymous or not, but suppose I want to lose matrix for word similarity or word distance. So, what are the different things in wordnet I can use? So, in wordnet, there are many other relations also defined. So, I can also use the fact that whether there is any other kind of relation between these two words or do the words, do these words share a common hypernym, hyponym and so on also you might have a look at the gloss of these two entries whether their glosses are also very similar.

So, I will say two words are similar if they share many different features of meaning and its features can be captured in terms of how many relations are common, how many words are common in their gloss and so on. Now, one thing that we must keep in mind when we are talking about establishing relation between different entities in wordnet, we are talking about relation between different synsets, and not the words. Because a word might have multiple synsets, and the relation that is there in one sense of word 1 and one sense of another word may not be there in the other synsets.

(Refer Slide Time: 12:59)



So, for example, is a word bank, this is sense one this is in the sense of you can say in the economics; and sense two is like river bank. And then I have another word like fund and there is s 1 that is for economy. So, I will say that this sense one of bank is connected to this particular sense of fund, but I cannot say that this is connected to fund, this is not true. So, I cannot say that bank is connected to fund saying that will not be correct, what I will say sense one of bank is connected to sense one of fund. So, we will talk about relations between synset of words and not word directly, all though we will try to extend it later to the words. So, instead of saying bank is like fund, I will say something like bank 1 is similar to fund 3, suppose that is the third sense of fund; and bank 2 is similar to slope 5 - the fifth sense of slope and so on.

Now, we will also compute similarity over words and synsets both. So, let us see how do we do that by using the wordnet hierarchy and any other information that we have. So, now this is something that that we have talked about earlier that if I want to find out similarity between words there are two very popular methods one is by using distribution algorithms and that we discussed in detail in the last week. There I can find out the distribution patterns of two words and compare those, but if I want to use a lexical resource like wordnet.

(Refer Slide Time: 14:52)

*Two classes of algorithms*

*Distributional algorithms*  
By comparing words based on their distributional context in the corpora

*Thesaurus-based algorithms*  
Based on whether words are "nearby" in WordNet

Pawan Goyal (IIT Kharagpur)      Lexical Semantics - WordNet      Week 8, Lec. 7      27

So, here can I use the idea that some words are more near in the wordnet hierarchy than others. So, if two words are near in the hierarchy, I might say they are similar; if they are very far apart in the hierarchy, they might be different. So, can I use this idea to establish if two

words are similar by using the wordnet resource? So, we will now see lot of such methods of doing that.

(Refer Slide Time: 15:21)

The slide has a blue header bar with the title "Thesaurus-based Word Similarity". The main content area contains a bulleted list:

- We could use anything in the thesaurus:
  - Meronymy, hyponymy, troponymy
  - Glosses and example sentences
- In practice, "thesaurus-based" methods usually use:
  - the is-a/subsumption/hypernymy hierarchy
  - and sometimes the glosses too
- Word similarity vs. word relatedness
  - Similar words are near-synonyms
  - Related words could be related any way
    - \* car, gasoline : related, but not similar
    - \* car, bicycle: similar

At the bottom of the slide, there is a navigation bar with icons for back, forward, and search, along with the text "Pawan Goyal (IIT Kharagpur)", "Lexical Semantics - WordNet", "Week 8, Lecture 2", and "10 / 27".

So, as such I can use any relations like meronymy, hyponymy, troponymy glosses examples, yes, but in particular the thesaurus-based methods that we have they mainly use the EJ hierarchy tree the hyponymy, hyponymy relation tree. Sometimes we will also use the glosses of the words. So, we will start by seeing some examples or some particular methods that try to use the hierarchy - wordnet hierarchy to capture the similarity between the words and then we will also see a particular method that uses the glosses of the different synsets to capture their similarity.

Now, so one thing that you might have seen or have understood by now that by using all these methods we are not capturing the synonymy as such we are not seeing that that we are finding two words that are very, very similar. What you finding is that two words that are related or are used in similar sort of context and topic. So, like if I take car and bicycle, they are quite similar, but car and gasoline they might be related, but not similar. So, by these methods car and gasoline might come closer, but all that means is that they are related they may not be exactly similar.

(Refer Slide Time: 16:47)

*Path-based similarity*

*Basic Idea*

- Two words are similar if they are nearby in the hypernym graph
- $\text{pathlen}(c_1, c_2)$  = number of edges in shortest path (in hypernym graph) between senses  $c_1$  and  $c_2$
- $\text{sim}_{\text{path}}(c_1, c_2) = \frac{1}{1 + \text{pathlen}(c_1, c_2)}$
- $\text{sim}(w_1, w_2) = \max_{c_1 \in \text{senses}(w_1), c_2 \in \text{senses}(w_2)} \text{sim}(c_1, c_2)$

Pawan Goyal (IIT Kharagpur)    Lexical Semantics - WordNet    Week 8, Lecture 1

So, now coming to the methods to capturing similarity across words, what is the first idea? First idea is to use the path between two words in the hypernym graph. And what can be a simple measure, I will say that two words are similar if the path that connects the two words in the hierarchy is small. So, two words are similar if they are nearby in the hypernym graph and to give a formal measure or quantity to that, I can define the path length between two concepts. So, path length between two concepts what will be that that is the number of edges in the shortest path in my graph between synsets c 1 and c 2.

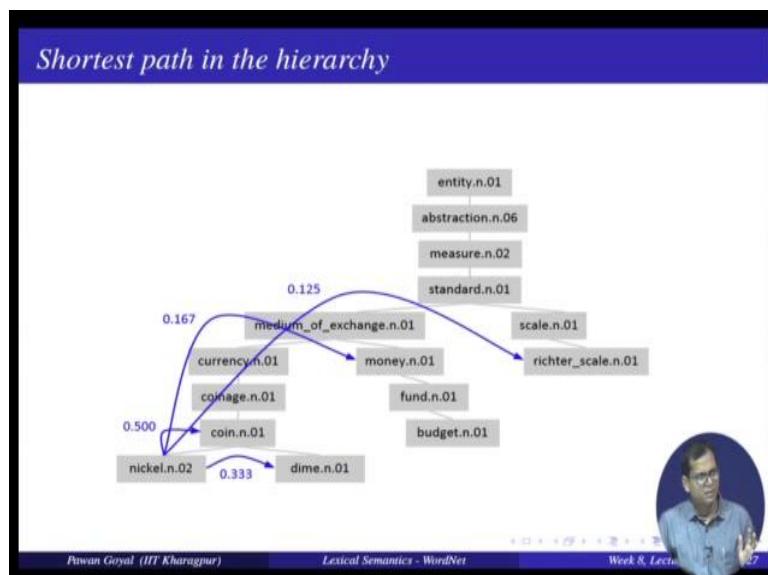
So, what is the length of the shortest path that connects c 1 and c 2 in my whole graph? Now, once have found this path length how do I use that to compare the similarity between these two concepts, so that is path length is large they are less similar; if path length is small they are very, very similar. So, my similarity should be inversely proportion to the path length. So, one measure can be 1 divided by 1 plus path length and this is one simple measure that is used. The path similarity between two concept is 1 divided by 1 plus path length of c 1 and c 2, so that is how we can find out the relation between two synsets.

So, now suppose I want to extend that to find the similarity between two words. So, one way is to find out similarity between all the synsets and take an average, but a more commonly accepted measure is find out similarity between all the possible pairs of synsets between the two words and take the maximum. So, what you mean by that? Suppose I have word 1, word 2 that is sense s 1 1, s 1 2, s 1 3 and this a synsets s 2 1, s 2 2. So, by using this measure, I can

find similarity between any two pairs  $s_1 s_1$ ,  $s_2 s_2$ , what is the similarity. Similarly, I can do for all these pairs.

Now, how do I establish similarity between  $w_1$  and  $w_2$ , so that is similarity is the maximum value of similarity between  $s_1 i$  and  $s_2 j$ . You take any sense for the first word; any sense for the second word whatever the maximum similarity between a pair gives me the maximum similarity or the similarity between these two words that is a very simple way in which I can extend all these ideas to word similarity. So, in whatever we will see the next methods, we will also always talk about similarity between synsets and extended for similarity between words. So, I will say similarity between words  $w_1$  and  $w_2$  is nothing but the maximum similarity between any of the synset of the word 1 and word 2.

(Refer Slide Time: 20:20)



So, let us take an example that how what will this pathway similarity look like. So, this is my wordnet hyponym graph. So, not all the notes are shown only if very few notes are shown. So, we are starting with entity, abstraction, measure, standard although with entity there will other concepts here. Then you are coming to a particular branch with where you have medium of exchange, currency, coinage, coin, nickel and so on.

Now, I want to find out similarity across two different concepts. So, what is similarity between nickel and coin? So, I will say what is the path length; path that connects nickel and coin, what is the length of this path. So, here the length is only 1. So, similarity between these two concepts will be 1 divided by 1 plus 1, so 0.5. And what is the similarity between nickel

and dime, it will be 1 divided by 1 plus path length and path length is 2, so it will be 0.33. And if you see nickel in a very different concept like Richter scale, so here you will find similarities 0.125, because the path length is 7. So, like that I can capture the similarity between two concepts by using the path length.

(Refer Slide Time: 21:29)

$$sim_{LC}(c_1, c_2) = -\log(pathlen(c_1, c_2)/2d)$$

Now, there is another similarity measure along the same lines. So, this is called L-C similarity. So, what it says, the similarity between two concepts is minus log of path length divided by 2 d. So, this is just a different function over path length. So, earlier we had a function 1 divided by 1 plus x, now they have function minus log x divided by 2 d. And what is d here, d is the maximum depth in my hierarchy. So, starting from the root node, what is the maximum depth of a hierarchy for any of the leaf note? And this helps in that this path length will always be less than or equal to two d and this will give me a similarity between these two concepts.

So, what is the problem with this L-C similarity or the previous similarity that we have seen? So, what they are saying for any two concepts find out the path length and the similarity is nothing but a function of path length if path length increases the similarity decreases. But one problem with these approaches is that any two pairs of concepts, if the path length is same, the similarity will be the same irrespective of wherever they occur in the tree.

So, let us just go back to the previous tree. So, what these approaches will say. So, what is similarity between coin and nickel? The path length is 1, similarity is 1 divided by 1 plus 1 that is 0.5. But what would be the similarity between entity and abstraction their path length is also 1, so their similarity will also become 1 divided by 1 plus 1, so that is 0.5, but ideally what would we want do we want the similarity of entity of entity abstraction to be the same as between coin and nickel. So, if you think about it as we are going down in the hierarchy, we are moving to very, very specific concepts. So, while we are moving this specific concepts, the same path length should amount to a higher similarity noun that it was doing earlier.

So, entity and abstraction this similarity should be much lower than the similarity, this similarity should be very very high, but these methods as of now as of now do not capture this idea. So, they will have this path length to contribute same way as this path length. So, now can we do something different? So that here the similarity becomes high, but here similarity becomes low. So, you want a matrix that let us assign different lengths to different average. So, what is the idea that we will be using?

(Refer Slide Time: 24:35)

*Concept probability models*

*Concept probabilities*

- For each concept (synset)  $c$ , let  $P(c)$  be the probability that a randomly selected word in a corpus is an instance (hyponym) of  $c$
- $P(\text{ROOT}) = 1$
- The lower a node in the hierarchy, the lower its probability

*Estimating concept probabilities*

- Train by counting "concept activations" in a corpus
- Each occurrence of *dime* also increments counts for *coin*, *currency*, *standard*, etc.

Pawan Goyal (IIT Kharagpur)    Lexical Semantics - WordNet    Week 8, Lecture 2    14 / 27

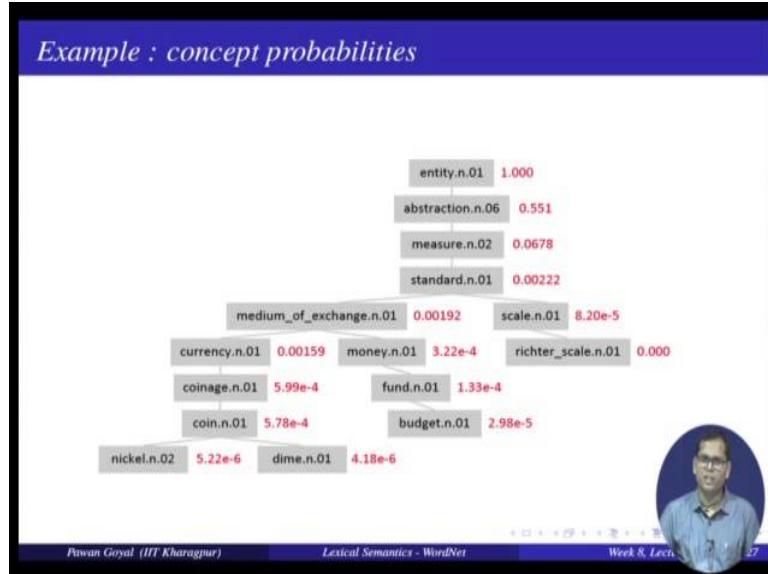
And for that we use the idea of concept probability model. Now, what is this? So, in the wordnet, whatever concepts we are seeing, we will assign them into probability. So, what is the probability with which I see this concept in a corpus? Now, idea would be whatever I am seeing in my corpus is an entity, because it is part of the tree where entity is the root. So,

whatever word is in the tree is an entity. So, in the other word, whatever word I am saying in the corpus is an entity, but it may not be an abstraction. So, there will be some words that are abstraction and some words that are not. So, what I will do whenever I encounter a word, I will find out what are all the concepts to which it contributes, and I will add a count to all these concepts. And finally, I will convert them to probability values.

So, what would happen? The root node will get a probability of 1, because everything I see is an entity, but as you go down these values will keep on decreasing. I will use this idea to convert them into log values and then taking the difference between the two values as the path length and that can converted to finding the similarity between two synsets. So, let us say  $P_c$  is the probability that a randomly selected word in the corpus is an instance of  $c$ . So, what would happen? Probability of root is 1, and a lower a node in the hierarchy, the lower is its probability. And how it can be estimated these probabilities, we count something called concept activations in the corpus. So, what would happen? Whenever encounter a dime, I will also increment a call for coin, currency, standard etcetera.

So, what is the idea? So, I have a wordnet hierarchy tree starting from root that is my entity and going down. So, what would happen? Suppose this is my word  $x$ , whenever I encounter this word  $x$  in my corpus, I increment its count by 1, and all its parents, because whenever I am encountering, I am also encountering this concept and so on. So, what would happen? Whatever I encounter I always add 1 to the root, but only to its parents. So, when I do that root will have, so all the counts will be added to root, and I can find the probability by dividing everything by the count of root.

(Refer Slide Time: 27:34)



So, suppose I do that on my corpus. So, this is one example. So, here you can see there are 1.9 million roughly instances overall because entity has been always appended with one, but there the numbers are different. So, diamond and nickel has only 8 and 10; coin has 1,108. So, you see, we are not showing the whole tree that is why there will some other branches of coin also that are not here. Now, once I have got these counts, I know opt in the probability values by dividing everything by this number. So, this will be my concept probability.

Now, how do I use this concept probability to define or define my edge weights? So, what I will do? I will convert them in some information value. What is the information content of each concept and the information content can be directly obtained by the probability values by using minus log probability. Idea is that if the probability of something is very high, it does not have much information, but the probability is low it contains a lot of information.

(Refer Slide Time: 28:47)

### Information content

*Information content*

- Information content:  $IC(c) = -\log P(c)$
- Lowest common subsumer :  $LCS(c_1, c_2)$ : the lowest node in the hierarchy that subsumes (is<sup>a</sup> hypernym of) both  $c_1$  and  $c_2$
- We are now ready to see how to use information content ( $IC$ ) as a similarity metric.

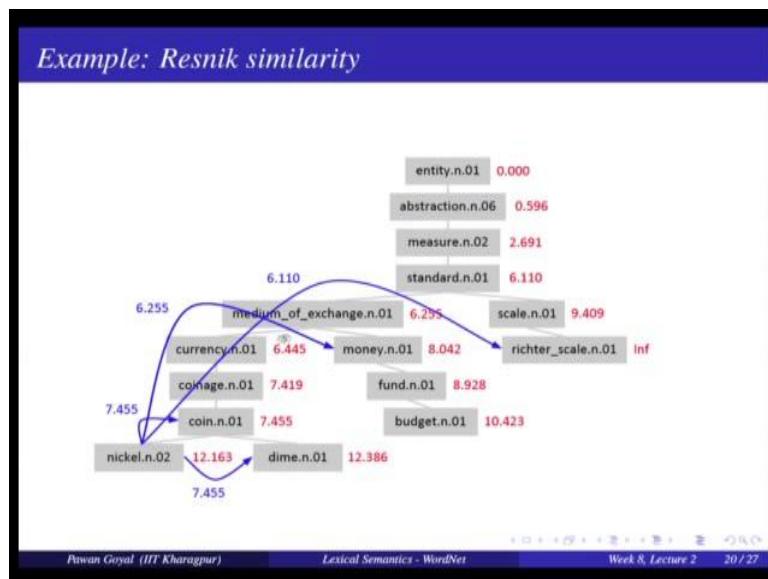


Pawan Goyal (IIT Kharagpur)    Lexical Semantics - WordNet    Week 8, Lecture 2 / 27

$$IC(c) = -\log P(c)$$

So, I use the information content of a concept as a minus logarithm of the probability of that concept. So, I can do that for all the concepts, and I can also define what is my lowest common subsumer that is if I take two nodes or two concepts  $c_1$  and  $c_2$ , the lowest common subsumer is the lowest node in the hierarchy that subsumes both the nodes or what is the lowest node in the tree where these two concepts meet.

(Refer Slide Time: 29:31)



And now we will see how you can use that computing similarity between concepts. So, these by using from the probability, if I take minus log probability, the other different numbers I will get. So, minus log 1 becomes 0, if entity is 0 then 0.5 minus 5 and so on. Now, what is something that you are seeing here? So, as you are going down these numbers are increasing. So, one simple way of capturing similarity between two words is by seeing what is the lowest common subsumer and what is the information content of that. So, nickel and dime, what is the similarity, path length is 2, but the lowest common subsumer that is coin has information content of 7.455. On the other hand, if I take these two concepts medium of exchange and scale, their common lowest subsumer standard has an information content of 6.11.

So, immediately you can see this can be said to have a higher similarity than this pair. So, what are the formal method by which we can capture this? So, one is Resnik similarity. So, it says how similar two words are depends on how much they have in common, so that is find out their lowest common subsumer and find out their information content of that and that will denote the similarity between these two concepts. So, it measures the commonality by the information content of the lowest common subsumer. So, like here nickel and dime, the similarity would be the information content of the L-C s that is coin. So, similarity between them is 7.455. Now, nickel and money similarity would be the information content of their L-C s that is medium of exchange, so that would be 6.255 and so on.

So, now, immediately you can see that as you keep on going up in the hierarchy, the similarity will be decreasing. So, it will be highest when you have took single leaf nodes, but if you keep on going up the similarity will decrease. So, this capturing what you wanted to do, but still there is one problem here. So, can you find out what is the problem? So, one problem here is if I find the similarity between coinage and money, this would be same as similarity between coinage and budget, yes, because their L-C s is the same. So, here what is being captured is that how much information they share, but what is not being captured is how much information they do not share or how much they are different. So, we have a different measure for capturing how much information they do not share.

(Refer Slide Time: 32:23)

*Proportion of shared information*

- It's not just about commonalities - it's also about differences!
- **Resnik:** The more information content they share, the more similar they are
- **Lin:** The more information content they don't share, the less similar they are
- Not the *absolute* quantity of shared information but the *proportion* of shared information

$$sim_{Lin}(c_1, c_2) = \frac{2\log P(LCS(c_1, c_2))}{\log P(c_1) + \log P(c_2)}$$

The information content common to  $c_1$  and  $c_2$ , normalized by their average information content.

Pawan Goyal (IIT Kharagpur)    Lexical Semantics - WordNet    Week 8, Lecture 2    21 / 27

$$sim_{Lin}(c_1, c_2) = (2\log P(LCS(c_1, c_2))) / (\log P(c_1) + \log P(c_2))$$

And this is called Lin-similarity. So, it says that this measure is not about just commonalities we also have to capture the differences. So, the more information they share, the more similar they are, but the more information they do not share the less similar they should be. So, accordingly the Lin-similarity between two concepts is defined as two times logarithm of or two times information content of the L-Cs divide by information content of the concept plus information content of the concepts. So, while doing that what would happen? Now if I take the similarity between coinage and money this would be 6.255 times to divide by 7.419 plus 8.042.

And if I take the similarity between the coinage and budget it will have a term of 10.423 in the denominator instead of 8.0242. So, immediately this similarity will become lower than this similarity. So, Lin-similarity is a much more well accepted measure than the previous measure that we have seen the Resnik similarity.

(Refer Slide Time: 33:41)

### Jiang-Conrath distance

*JC similarity*

We can use IC to assign lengths to graph edges:

$$dist_{JC}(c, \text{hypernym}(c)) = IC(c) - IC(\text{hypernym}(c))$$

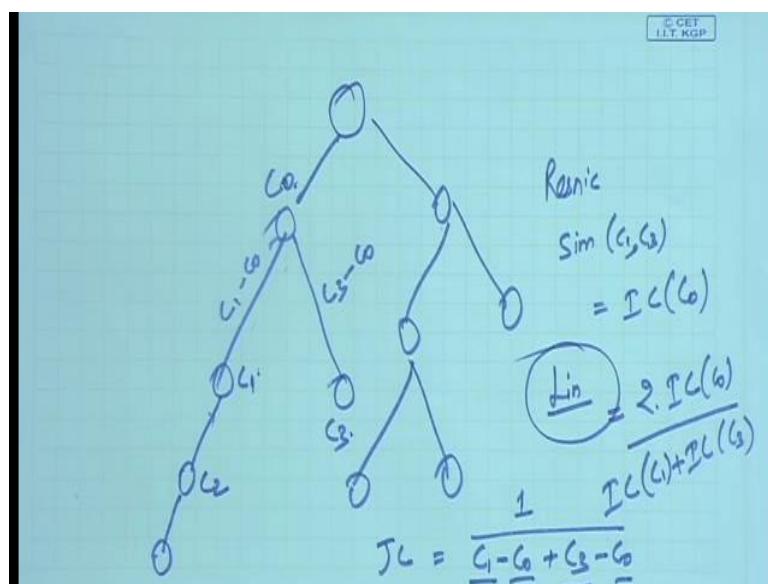
$$\begin{aligned} dist_{JC}(c_1, c_2) &= dist_{JC}(c_1, LCS(c_1, c_2)) + dist_{JC}(c_2, LCS(c_1, c_2)) \\ &= IC(c_1) - IC(LCS(c_1, c_2)) + IC(c_2) - IC(LCS(c_1, c_2)) \\ &= IC(c_1) + IC(c_2) - 2 \times IC(LCS(c_1, c_2)) \end{aligned}$$

$$sim_{JC}(c_1, c_2) = \frac{1}{IC(c_1) + IC(c_2) - 2 \times IC(LCS(c_1, c_2))}$$

Pawan Goyal (IIT Kharagpur)      Lexical Semantics - WordNet      Week 8, Lecture 2      23 / 27

There are some variations here. So, for example, the JC similarity, what they say in JC similarity find out or give a value to each h that is a distance between two concepts. And this would be the information content of the concept minus the information content of the hyponym, and I can define the distance between two concepts as their distance. So, how do I go from one concept to its L-C s, and second concept to its L-C s and I just add the distances. And I compute similarity by taking the inverse of this distance.

(Refer Slide Time: 34:27)



So, what am I doing here I am having different nodes in my hierarchy yes and suppose that you have already found, what is their information content. So, what do you do in the case of Resnik similarity, in Resnik similarity the c 1, c 2, c 3 and this is c 0; in Resnik similarity the similarity of c 1 and c 3 is nothing but the information content of c 0. So, as per Resnik similarity of c 1, c 3 is  $IC(c_0)$ . As per Lin-similarity, this would be two times information content of c 0 divide by information content of c 1 plus information content of c 3.

Now, in JC similarity, what you would do? You would count you would find out the distance this is distance is nothing but  $c_1 - c_0$ , and this distance is  $c_3 - c_0$ . So, for JC similarity it will be  $c_1 - c_0 + c_3 - c_0$  is the distance between these two concepts. And the similarity between the 1 divided by that. And that is what is written here? Information content of c 1 plus information content of c 2 minus 2 times information content of the L-C f of the 2 and that is what you can say here  $IC(c_1) + IC(c_3) - 2 \cdot IC(LCs)$ . So, I hope by this example you understand what is the difference between Resnik similarity, Lin similarity and JC similarity; and among these Lin-similarity is very, very popular.

So, I hope it is clear that how do you apply these three different similarity measures. So, here you have example that if we use the JC similarity, what is the different values that you will obtain among different concepts. So, I will encourage that you will you try and find out that say suppose between nickel and Richter scale or between nickel and coin can you obtain the same values by applying the formulas.

(Refer Slide Time: 37:03)

The (extended) Lesk Algorithm

- Two concepts are similar if their glosses contain similar words
  - Drawing paper: **paper** that is **specially prepared** for use in drafting
  - Decal: the art of transferring designs from **specially prepared paper** to a wood or glass or metal surface
- For each n-word phrase that occurs in both glosses, add a score of  $n^2$
- paper** and **specially prepared**  $\rightarrow 1 + 4 = 5$

Pawan Goyal (IIT Kharagpur)      Lexical Semantics - WordNet      Week 8, Lecture 1

So, now, I will also talk about briefly the other approach for computing similarity between two different concepts in wordnet. So, till now we have only used the hierarchy tree in wordnet, and I am saying two words are similar, if they are nearby in the hierarchy tree or some other formulation and that we have seen some examples. Now, suppose I want to use their glosses, the way the different concepts are defined in wordnet for comparing similarity. So, these are very simple algorithm called Lesk algorithm, also have a extended Lesk version that is used for that. And what is the idea two concepts are similar if their glosses contain similar word. So I have the word drawing paper defined as paper that is specially prepared for using drafting; and decal - the art of transferring designs from specially prepared paper to a wood or glass or metal surface.

I want to find out how similar they are. So, what I will do? I will see how many words are common there. So, as such you are seeing that three words that are common paper, specially and prepared that occur in both the glosses. So, what algorithm does is that it counts how many n-grams are common, so n-gram in the sense of one unigram, bigrams, and trigrams and so on. So, you will see that this bigram is specially prepared is common to both the glosses and the unigram page is common. So, whenever an n-gram is common, it adds a score of  $n^2$ , so that you have is given for a commonality of bigram trigram and so on. So, in this case, what would be the similarity one bigram is common, so two square and one unigram is common and one square. So, similarity would be  $2^2 + 1^2 - 5$ , 1 plus 4 - 5 that is the similarity of using Lesk algorithm.

(Refer Slide Time: 39:04)

The slide has a blue header bar with the text "Problem in mapping words to wordnet senses". Below the header is a white area containing the sentence "I saw a man who is 98 years old and can still walk and tell jokes". At the bottom right of the slide is a circular profile picture of a man with glasses and a blue shirt. The footer of the slide includes the text "Ptwan Goyal (IIT Kharagpur)", "Lexical Semantics - WordNet", "Week 8, Lecture 1", and a navigation icon.

So, we have talked about what are the different relations we can capture using wordnet, and we have also seen how we can find out similarity of across two words, and that looks like very simple method once the wordnet tree is given. And you might also wonder this might be a better method of capturing similarity than distribution similarity, because I have a manually created this orders, I know which of the words occur where in the tree I can simply use the distance or measure to find out how similar they are. But there is one inherent problem in using wordnet for any of the task. So, can you think of what is the problem?

So, let me give you the hint if you know if you want to capture similarity across synsets wordnet is very good, it can capture the similarity between the synsets very nicely but if you want to catch similarity between words that is very difficult. Now, when we encounter natural language, we will only encounter the words; and when we see the words, we do not know what are the synsets that are being used. So, I cannot directly apply wordnet there, because I do not know the sense, I can only do an approximation where I can find out ok, I am assuming this word corresponds to or I am applying the methods I used for synsets for the words. This would be an approximation. And it works sometimes, but does not work some other times.

So, to be able to apply wordnet, to be able to use wordnet, one important problem that we have to deal with is I need to find out if a word is used in a particular sentence what is the wordnet sense that has been used for that and that is difficult problem that we will try to

address. But this problem is very, very common and we will just take a very simple example for that. So, you see very easy sentence, I saw a man who is 98 years old and can still walk and tell jokes. Yes, this is a very simple sentence. Now, for the simple sentence, what do you think, are there many different synsets of this word, or this whole sentence, or there is one interpretation. So, when we hear this term this sentence we have only one interpretation in mind, but in wordnet what are different synsets of individual words.

(Refer Slide Time: 41:44)



So, let us see. So, if I go to wordnet, the word saw has 25 senses, man has 11 senses, years has 4, jokes has 4, tell has 8 and so on. So, now, if you combine all these together, they are has a 67 million plus senses that are possible sentence this might look like a very extreme case, but you might have examples where they are multiple divisions for the same sentence, and different words can have can occur in multiple synsets. So, my problem is if there are so many synsets, how do I find out what exact synset wordnet is being used, and that is where we talk about the problem of word sense disambiguation among the many possibilities disambiguate the particular sense of the word, and that we will start in the next lecture.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 39**  
**Word Sense Disambiguation – I**

Welcome to the third lecture of this week. So we are talking about lexis semantics and there we dealt with what are the various relations we can establish between the word forms. So we were talking specific in traditional now that are lexemes different items in my lexicon. And we also talked about word net. That what are the different relations we can establish between various lexemes word net. And what are the different standard forms like using sins at and what is the hierarchical award net, how do you use the define word similarity and so on. And finally, we ended up the saying that there is one in head problem engaging word net that is when you get a new sentence you do not know what is the particular sense of the word that is being used in the sentence. So you cannot apply the methods vary directly.

So that is fair. So we deal with this problem that given a sentence for each word find out what is the sense in the word net that is being used here. And this problem is called versions disambiguation. So very classical problem in NLP and lot of research has happened in over last few decades, so we will talk about what are the most important methods for dealing with this problem. And we will talk about both simple approaches and machine learning based approaches and also some unsupervised approaches. So we will cover that in the next 2 lectures starting today. So this is our topic that is word sense disambiguation.

(Refer Slide Time: 01:55)

**Word Sense Disambiguation (WSD)**

**Sense ambiguity**

- Many words have several meanings or senses
- The meaning of **bass** depends on the context
- Are we talking about music, or fish?
  - An electric guitar and **bass** player stand off to one side, not really part of the scene, just as a sort of nod to gringo expectations perhaps.
  - And it all started when fishermen decided the striped **bass** in Lake Mead were too skinny.

**Disambiguation**

- The task of disambiguation is to determine which of the senses of an ambiguous word is invoked in a particular use of the word.
- This is done by looking at the context of the word's use.

Pawan Goyal (IIT Kharagpur) Word Sense Disambiguation - I Week 8, Lecture 3 2 / 15

So let me just define the problem again. So we have seen in this week that there are many words that are having several meanings. And then we can call them different senses of the same word. So now, example can be of the word bass. So you can use bass or base depending on that they you want to use that in the context of fish or music. Now the problem is whenever you are given this word I will be talking about music or fish, you know. So let us see these 2 sentences here. The first sentence is an electric guitar and best bass players stand off to one side not really part of the scene just as a sort of nod to gringo expectation perhaps. This is one sentence. And the second sentence is talking about some trip bass in Lake Mead were too skinny. So now, the same word is being used in both the sentences. And I have to find out whether the word is being used the sense of fish or music. Now once you see the sentences, how do you find out that what is the meaning of this word in the sentence. You will try to look around the context in what context this word is being talked about. Accordingly, you will choose one sense over and another one. Now this whole the field of words has disambiguation.

So this deal with this problem, that I have to find out the sense of the word depending the context know. How can I do that computational? So we can define the task of the disambiguation that is to determine which of the senses of an ambiguous word is invoked in a particular use of a word. So we say we are doing this problem only for the

ambiguous words. Whenever it has more than one senses in a particular use of that word, what is the sense that has been used.

Now, and as you can see we will deal with this problem by looking at the context of the words use.

(Refer Slide Time: 04:01)

The slide has a dark blue header bar with the word 'Algorithms' in white. The main content area is white with a black border. At the bottom, there is a dark footer bar with the names 'Pawan Goyal (IIT Kharagpur)' and 'Word Sense Disambiguation - I', the text 'Week 8, Lecture 3', and a page number '3 / 15'. The main content is a bulleted list of approaches:

- Knowledge Based Approaches
  - Overlap Based Approaches
- Machine Learning Based Approaches
  - Supervised Approaches
  - Semi-supervised Algorithms
  - Unsupervised Algorithms
- Hybrid Approaches

So now how exactly we do that? There are many different algorithms that handle this problem. So there are approaches based on their knowledge base approaches that use different overlap based methods, for handling that. Then there is some machine learning based approaches they use supervised approaches unsupervised approaches and semi supervised approaches. And also there are some hybrid approaches. And in the literature there are a lot of different methods for solving this problem.

Now, what we will do? We will talk about some very basic methods. So that you have the intuition with you that what are the different features what are different methods you can apply for word sense disambiguation.

(Refer Slide Time: 04:39)

**Knowledge Based Approaches**

**Overlap Based Approaches**

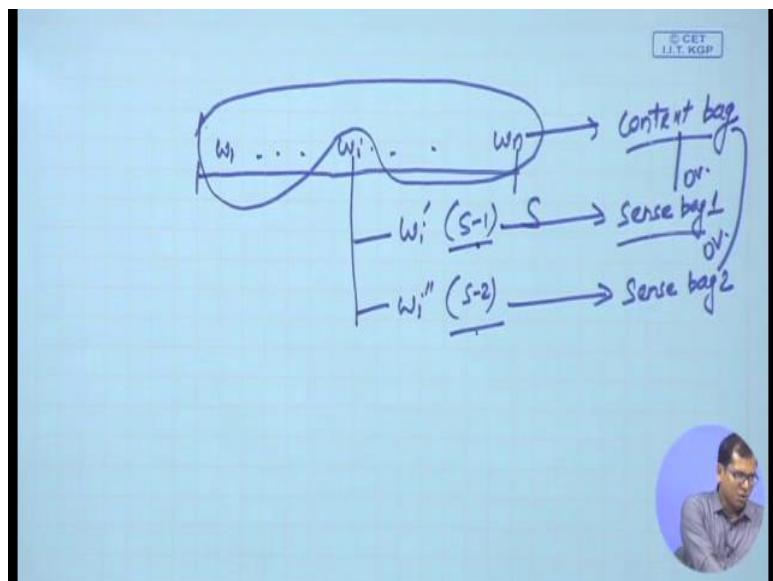
- Require a **Machine Readable Dictionary** (MRD).
- Find the overlap between the features of different senses of an ambiguous word (**sense bag**) and the features of the words in its context (**context bag**).
- The features could be sense definitions, example sentences, hypernyms etc.
- The features could also be given weights.
- The sense which has the maximum overlap is selected as the contextually appropriate sense.

Pawan Goyal (IIT Kharagpur)      Word Sense Disambiguation - I      Week 8, Lecture 3      4 / 15

So, now starting with the knowledge base approaches, so there are some sorts of overlap based approaches. They try to use some sort of overlap. Now how do they find is overlap? They would require a machine readable dictionary. So like word net is my machine readable dictionary. You can have some other sort of the torus also. That are in a machine readable form. So by machine readable I mean all the entries and all the information is such that you can easily access that and you can make use of that.

So what approaches will do? They will try to find the overlap between the features of different senses of an ambiguous word that is they will call it sense bag and the features of the word in its context, and call it as the context bag. And they will try to measure the overlap. So now, just to explain this idea let us see I am having a sentence.

(Refer Slide Time: 05:34)



S, that contains some words  $w_1$  to  $w_n$ . And the word  $w_i$  can be used in 2 senses.  $w_i$  prime and  $w_i$  double prime; it is sense 1, sense 2. Now in the sentence this word  $w_i$  would be used only for one sense and that is what we are assuming. This will be true in most of the cases unless, the speaker himself wants to imply 2 different meanings of the same words. So we will say in general for this word  $w_i$  you would have only one sense.

So now I want to find out depending on this context whether sense one should be taken off sense two should be taken. So idea would be for each sense I will construct the sense bag. And we will see how do we construct sense bag. So this I will convert that to a sense bag. Sense bag 1 and this I will convert to sense bag 2. Now looking at the context that is you can think of all the words that are coming in the sentence other than this word I will construct a context bag. Now I will try to find out what is the overlap between context bag in sense bag 1 and what is the overlap here. And I will take the answer as the as the one sense that is having the highest overlap. And this is to explain that simply, I have various senses, let us context the sense bag context that the context bag take the overlap take the sense with the maximum overlap.

Now, to construct this sense bags and context bag, you will use various features that are provided in the dictionary that you have. So features could be like what are the sense definitions that I am having, what are the example sentences that are provided, different

hyponyms and so on. And any other criteria that you can use from the dictionary and you will use that to construct both the sense bags and the context bag take the one with does maximum overlap.

(Refer Slide Time: 07:53)

The slide is titled "Lesk's Algorithm". It defines two bags:

- Sense Bag:** contains the words in the definition of a candidate sense of the ambiguous word.
- Context Bag:** contains the words in the definition of each sense of each context word.

Example text: "On burning **coal** we get **ash**".

Ash	Coal
<ul style="list-style-type: none"><li>○ Sense 1 Tree of the olive family with pinnate leaves, thin furrowed bark and gray branches.</li><li>○ Sense 2 The <b>solid</b> residue left when <b>combustible</b> material is thoroughly <b>burned</b> or oxidized.</li><li>○ Sense 3 To convert into ash</li></ul>	<ul style="list-style-type: none"><li>○ Sense 1 A piece of glowing carbon or <b>burnt</b> wood.</li><li>○ Sense 2 charcoal.</li><li>○ Sense 3 A black and <b>incombustible</b> substance formed by the partial decomposition of vegetable matter without free access to air and under the influence of moisture and often increased pressure and temperature that is widely used as a fuel for <b>burning</b></li></ul>

In this case Sense 2 of ash would be the winner sense.

Speaker photo: Pawan Goyal (IIT Kharagpur)

Navigation icons: back, forward, search, etc.

Now, let us take a simple example or in actual algorithm that that does that. So we have lesks algorithm. So we talked about this algorithm in the previous lecture also. For a different case when you wanted to find out what is some ready between 2 different senses. Now suppose I want to use this this less algorithm for the purpose of versus disambiguation. So what it does? So it constructs the sense bag context bag and how are they constructed? The sense bag contains the words in the definition of the candidate sense of the candidate part.

So I have a word that it ambiguous can have multiple senses, for each of the senses which I will construct one sense bag. And the sense bag is constructed by using the definitions that I have in my dictionary. And how do I construct the context bag? I take all the words in my context, take each of the senses of the context word and then take their definition all these together become my context bag. So I have a single context bag and I have multiple sense bag for different senses of the word. And then I take the overlap and this overlap is taken by using the leska algorithm similar to what we saw in

the previous lecture, that if there is a match of  $n$  a particular  $n$  gram I a d score of  $n$  square.

So now let us take this case. So I have a sentence on burning coals we get ash. Now hear the word ash it ambiguous and I want to find out what is the correct sense of this word that is usually sentence. And suppose for simplicity we are using only the word coal as my context. So what am I going to do? I will find out, I will make different sense bag for the word ash whatever senses are recorded in my word net, and I will construct the context bag by using all the definitions of the word coal. And then I will try to measure the overlap. So suppose we do that, so here is the case from by taking the dictionary definitions from word net. So ash has 3 senses. So one is corresponds to some trees second corresponds to some solid residue and third is to converting to convert that into ash. And we want to find out what which of the 3 senses is used here.

So what we will do? We will take the word coal that is the only context for we are using here, but in general you can use any number of context words. I take all it is sense definitions that are defined combine them together to make a context bag. So this whole thing together becomes my contacts bag. Now I measure it similarity with each of the senses. And if you do that we find the sense to 3 of the words are matching here and sense to become a winner.

Here, what you using here? You are also using that is stemming. So this is optional you might use stemming you may not use stemming. So this is the generic idea of leska algorithm. I hope this is clear, if I want to use multiple words I can repeat the same thing. I suppose I am using burning coal and get all the 3 words, I will take all their definitions and combine them together into a single context bag. And then measure the similarity of each of senses with this whole context bag.

(Refer Slide Time: 11:11)

**Walker's Algorithm**

- A Thesaurus Based approach
- **Step 1:** For each sense of the target word find the thesaurus category to which that sense belongs
- **Step 2:** Calculate the score for each sense by using the context words.  
*A context word will add 1 to the score of the sense if the thesaurus category of the word matches that of the sense.*
  - E.g. The money in this bank fetches an interest of 8% per annum
  - Target word: bank
  - Clue words from the context: *money, interest, annum, fetch*

	Sense1: Finance	Sense2: Education
Money	+1	0
Interest	+1	0
Fetch	0	0
Annum	+1	0
Total	3	0

Context words add 1 to the score of the sense when the topic of the sentence matches the category of the sense

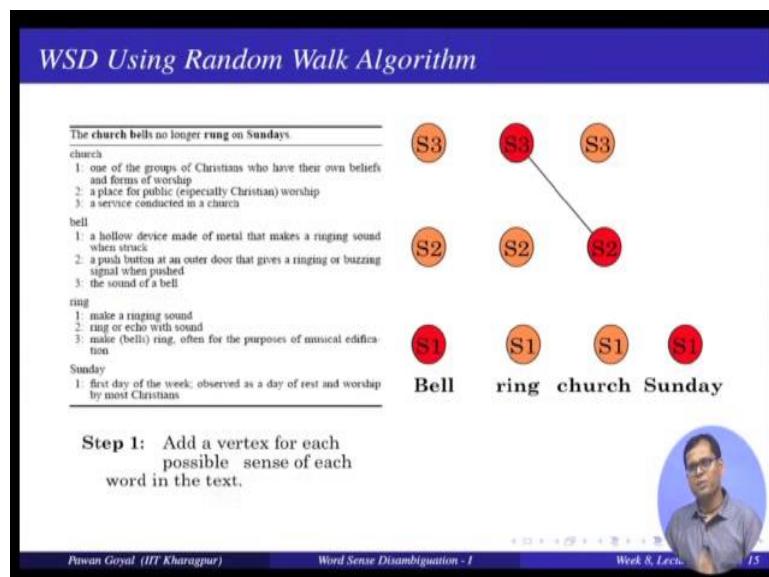
Pawan Goyal (IIT Kharagpur)      Word Sense Disambiguation - I      Week 8, Lecture 15

So now let us take another scenario, where we have a different kind of source. So what is this source now? For each word we are told what are the categories these senses it is senses belong to. So I have some finite set of categories. This can be like finance category, location categories, sports category and so on. And each word has been recorded as it has a sense in these this category. So now, the problem would be when I am given a sentence a word can have senses in multiple categories find out what is the a probably category in this sentence. So how do we use this framework here? So what I will do, for each sense of the target word if find out what is the source category to which this belongs; now calculate the score for each senses by using context words. No idea would be the context words again would be having the researches all categories. So for each context, but I will find out what is the source category over I will see by using the context word which thesaurus category getting the highest count.

So here, what we will be doing? A context word will add score of 1 of the sense if the thesaurus category of the word matches that of the sense. So let us take this sentence the money in this bank fetches interest of 8 percent per annum, and suppose my ambiguous word here is bank has 2 senses. One in the case of safe finance that is for the money bank that is also used in the sentence another could be river bank also can be used as location or something. And I take each context words, money interest and annum fetch. Each again would have their own dissolves category.

So what we will be doing? I will take all these words money interest fetch annum. I know bank have 2 different senses finance and location. So for each of these context words I will put a one if the match any of this, this all I can take as much. So money message finance, so I will put a one here, interest message finance again I will put a one here, fetch does not meet any of these senses. So both are 0 and a message finance again plus 1. And then I will add the score of both the senses and I in here I say that a finance the score is 3 for locations score is 0. So I will take the sense of bank is finance in this particular case. So these are simple approach it by just taking the dictionary definition thesaurus category.

(Refer Slide Time: 13:49)

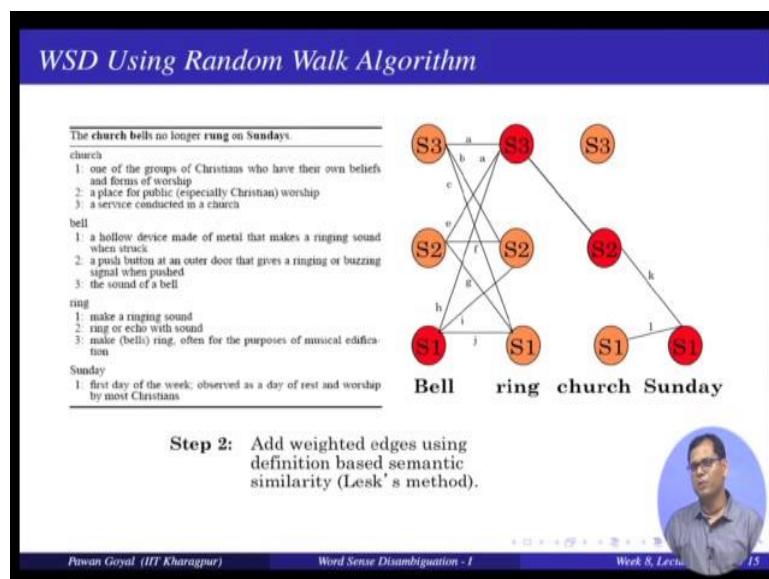


So there is one problem in both these approaches. That is work for disambiguating the sense of a particular word we are not disambiguating the sense of the other context words. We are taking all the possible senses of them into a single context bag, but suppose you want to solve this problem where in the same sentence there are multiple ambiguous words and you want to disambiguate each of them together. Then can we do that? And this can be done by making each of the fact that although each word might have multiple senses, but the particular senses that are being used in this sentence there will be somehow connected to each other.

So we want to exploit this and for this we can use some sort of grab based method. Similar to what we do in the case of PageRank algorithm. So let me just quickly explain this idea. And we will actually deal with this PageRank algorithm in detail when we talk about summation application, but I will try to give you the intuition. So this is the problem that suppose I have the sentence, the church bells no longer ring on Sundays. And I can say there are 4 words bell rings church in Sunday. And I want to disambiguate the senses of these words.

And also suppose that from the dictionary I know what are there sense definitions. So here the word church has 3 senses, bell has 3 senses, ring has 3 senses, and Sunday has one sense. And I want to disambiguate the first 3 words together. So how do I do that? So idea would be that is treated as a graph based problem, where for each word I first write down what are different senses of this word. So I have made a vortex for each of the sense of these words. So there are 3 words is for a bell 3, for ring 3, for church and one for Sunday. So I have now 10 words in this graph. Now what will be the idea? I will now try to connect the different words together by seeing what is the overlap among different sense definition.

(Refer Slide Time: 16:05)



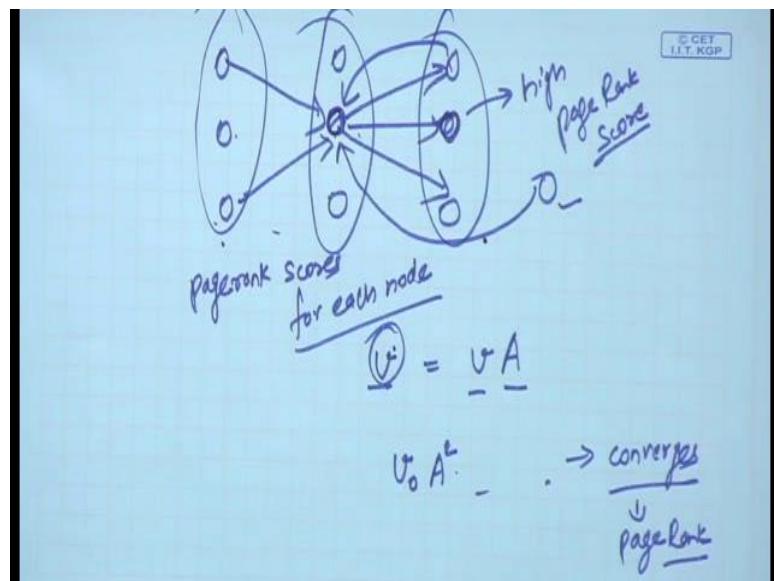
So I will now try to connect some weighted edges between different sense definitions by using Lesk's method. So the Lesk's method is by similar by simple for finding similarity

between 2 sentences. I use that find out what is the similarity between any 2 sense definitions, and add a weight between these 2. And this will give me some sort of matrix like that. So I have all the sense definitions connected. Now one thing one important thing is that for the same word I will not have to connect it is own sense definition. So there will not be any edge between S3 and S2 for the same word bell, but they will be edges between different words. So S3 of bell an S2 of ring would be connected by what is the similarity between these 2 sense definitions.

So idea is that now we want to somehow find out that particular combination here. So one sense for each of the 3 words such that their overall similarity is the highest. So they are having the highest similarity. And one particular method that can use for that is using it a PageRank kind of algorithm. So idea would be, if there are multiple, if there are appropriate sense definitions that are similar to each other, if I use the PageRank algorithm, they would have very high ranking school because they will all contribute to each other.

So what do we do? So for this graph once we have all the word decease we have all the edge bit also, we treated as a problematic fear finding the PageRank for each vertex of this graph and to give you simple intuition.

(Refer Slide Time: 17:56)

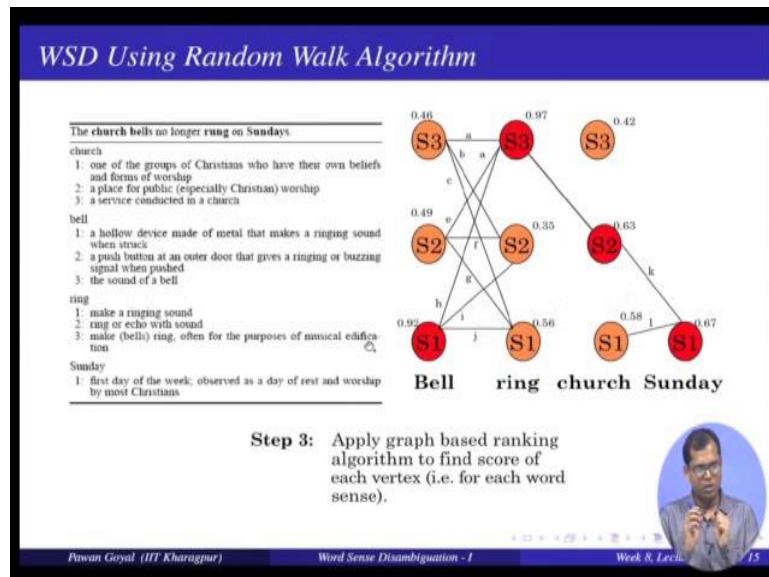


So now, what would we have? So we have 10 nodes the graph and I try to connect them by different numbers. These numbers depend on their lesks similarity. Now once I have this if I want to compute the PageRank scores for each node. And now what is the algorithm for computing paging is score if you take it the simple idea that is I suppose this PageRank it completed in a creative manner. So we start with some initial random is course, this will be some sort of probable distribution, and I use this equation  $v$  equal to  $v$  A to find out what will be the PageRank score and we denotes the PageRank scores. And how do we actually come up with this score? Start with some initial  $v_0$  and then keep on multiplying a square a cube until it converge and that becomes your PageRank score.

So we will talk about this algorithm in detail later and for now you can just quickly have a look at what is the PageRank algorithm, but the idea would be once we do all that you will find one PageRank score for each node. And the nodes that are having a lot of connections with other nodes and with high in degree connection will be given a higher PageRank score. So if suppose this node is connected to multiple sense definitions. It is connected to this sense definition, this sense definition, and this sense definition; that means, this might be one of the important senses in this for this particular word. And idea would be to find out for each word one sense that is having the highest connection and this would be someone that will get a high PageRank a score.

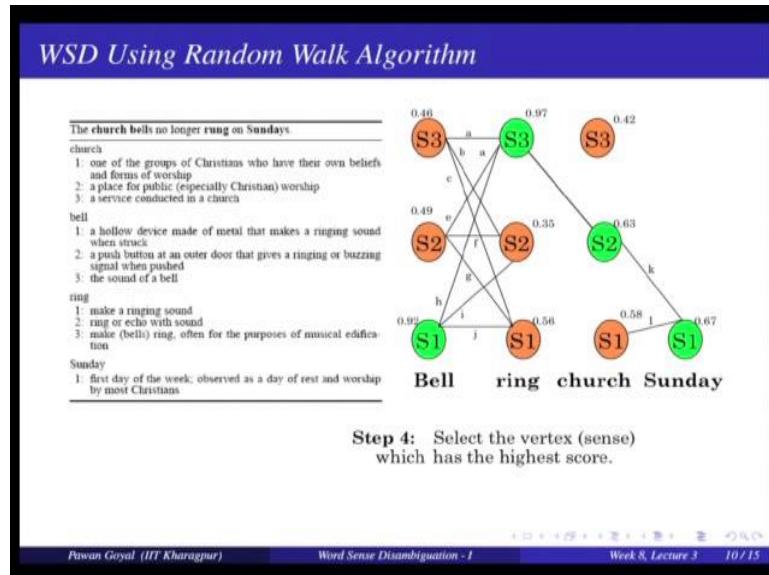
So I will give PageRank the score to each of the 10 node and then for each word I will pick the one that is having the highest among this set. So I will take one from each of the set and here I anyway choose this one. And this becomes my final disambiguated sense. So once I have added all these weighted adages, I will apply the graph based ranking algorithm, so this can be PageRank algorithm.

(Refer Slide Time: 21:00)



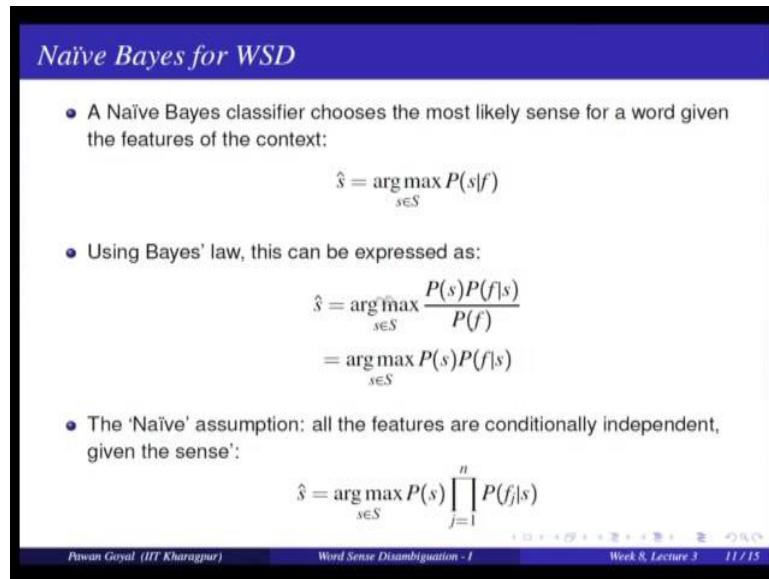
Now once we get all this course I will for each word I will choose the one with the highest score. An idea here is the sign the purpose sense that is connected to multiple other sense which for multiple words would get a higher score. And that is the intuition that I want to choose the particular sense that is having a similar definition has many other senses. And this I do for each of the word. So here suppose S1 in for bell, one S3 for ring and S2 for church are selected and they are my final disambiguated senses for these 3 words. And this is my overall algorithm.

(Refer Slide Time: 21:47)



Now, we can also use some machine learning based methods. And one simple idea would be to use a Naive Bayes algorithm.

(Refer Slide Time: 21:53)



$$\hat{s} = \text{argmax } P(s|f)$$

$$\hat{s} = \operatorname{argmax} P(s) \prod_{j=1}^n P(f_i|s)$$

So Naive Bayes algorithm we are using for doing classification here, what is the classification task for a given word there are multiple senses, from the sentence find out what is the appropriate sense. So this method has to be word specific and for each word you might have as many classes as the number of sense of this word.

(Refer Slide Time: 22:37)

The image shows a handwritten derivation of the Naive Bayes formula for word sense disambiguation. It starts with the expression for the probability of a word given a sense and context:

$$P(\text{Sense-}A \mid \text{collocation}) / P(\text{Sense-}B \mid \text{collocation}) = \operatorname{argmax}_s \frac{P(s) P(f|s)}{P(f)}$$

Then, it is shown that the denominator  $P(f)$  is common for all senses and can be ignored for finding the maximum. This leads to the simplified formula:

$$\operatorname{argmax}_s P(s) P(f|s)$$

Finally, it is shown that the product of probabilities  $P(f_i|s)$  for all features  $f_i$  is equivalent to the probability of the context  $s$ :

$$= \operatorname{argmax}_s P(s) \prod_{i=1}^n P(f_i|s)$$

A diagram on the right illustrates this with a node labeled  $s$  at the top, which branches down to nodes  $f_1, f_2, f_3, \dots, f_n$ , representing the features of the context.

So the problem here is, find out the sense of the word that gives the maximum probability as given  $f$  and we have a set of features that I will extract from the context around this word. So this for a given there can be multiple senses I want to find out the sense that is having this score at maximum. Now because Naive Bayes genetic model; that means, the sense comes on the off first and then the features are generated from their different features. So how do we compute this particular probability? This would be nothing, but  $\operatorname{argmax}_s$  probability  $f$  given  $s$  probability  $s$  divided probabilities  $f$  and because this common for all the sense this is  $\operatorname{argmax}_s P(s)$ ,  $P(f|s)$  is now  $P(s)$  nothing, but the prior probability of the sense. That is if a particular sense of the word is more commonly used

that many will get a high prior and  $P(f_i \mid s_i)$  what is the probability of different features being observed or generated by this sense.

So Naive Bayes model what we do? We make this assumption that each of these features are independent of each other given  $s_i$ , given the sense  $s_i$ . So suppose there are features  $f_1$  to  $f_n$ . So I was write it as  $\text{argmax } s_i P(s_i) \text{ is equal to } 1 \text{ to } n \text{ probability } f_i \mid s_i \text{ given } S$  and this I will do for each sense and I will take the one that is giving you the maximum score.

Now, what is important here is what are different features that you will be choosing for a Naive Bayes method. And these features have to come from the context around the word. So what are different things I can use in the context? I can use water different part of speech that are being used, what are different words and so on. Let us see for this task what are the important features. So this we have already seen.

(Refer Slide Time: 24:58)

### Training for Naïve Bayes

- $f_i$  is a feature vector consisting of:
  - POS of  $w_j$
  - Semantic and Syntactic features of  $w_j$
  - Collocation vector (set of words around it) → next word (+1), +2, -1, -2 and their POS's
  - Co-occurrence vector
- Set parameters of Naïve Bayes using maximum likelihood estimation (MLE) from training data

$$P(s_i) = \frac{\text{count}(s_i, w_j)}{\text{count}(w_j)}$$

$$P(f_i | s_i) = \frac{\text{count}(f_i, s_i)}{\text{count}(s_i)}$$

Pawan Goyal (IIT Kharagpur)      Word Sense Disambiguation - I      Week 8, Lecture 15

$$P(s_i) = \text{count}(s_i, w_j) / \text{count}(w_j)$$

$$P(f_i | s_i) = \text{count}(f_i, s_i) / \text{count}(s_i)$$

Some for this task, I can use the part of speech of surrounding words, and what are different semantic and syntactic features. I can use co-occurrence vector that is what are the different other words this sense is used with. And then I can use this collocation

vector that is water in general the next word, next-next word, previous word, previous to previous word and their part of speech tags, when the word is used in this particular sense. So I will construct this this side of feature vector by using this all these examples. And the 2 parameters of a model that is the prior probability of the sense and the probability of features given the sense can be computed from my corpus by using simply maximum likelihood estimate.

So how do we compute probability of the sense? S i, number of times the sensitive divide by number of times this word is used. Improbability of a particular feature to the senses sense number of times this feature is observe with the sense divide by number of times all other features are or number of times senses used. So these are simple formula for mle and you use that and plug into the algorithm and you have your Naive Bayes models 30 for use. And you can use some other approaches like decision list algorithm. So what is the idea?

(Refer Slide Time: 26:27)

### Decision List Algorithm

- Based on 'One sense per collocation' property
  - Nearby words provide strong and consistent clues as to the sense of a target word
- Collect a large set of collocations for the ambiguous word
- Calculate word-sense probability distributions for all such collocations
- Calculate the log-likelihood ratio
 
$$\log\left(\frac{P(\text{Sense} - A | \text{Collocation}_i)}{P(\text{Sense} - B | \text{Collocation}_i)}\right)$$
- Higher log-likelihood  $\Rightarrow$  more predictive evidence



Pawan Goyal (IIT Kharagpur)      Word Sense Disambiguation - I      Week 8, Lecture 15

So you will use this hypothesis of one sense per collocation. Now what is this one sense for collocation hypothesis? For a given word idea is that with the particular collocation it is use only in one sense. For example, take the word like bat. We saw there are 2 different sense. One is like it creates cricket bat and what another it like a flying mammal. Now suppose you take this collocation cricket bat. So the word cricket covering before

bat. With this collocation the word bat will always use in one sense only not the other sense. This is the idea. Can I get with the collocations to find out with this collocation this word to be with all in this sense and with the other collocations it will be used in another other senses. And this is called the one collocation per sense property. Sorry one sense per collocation property.

So how do I start about getting these set of collocations for an ambiguous word? So initially I can try out all the possible set of nearby words that occurs with this word more often. And once I have done that for each such collocation I can compute what is the probability of a particular sense being used for this collocation with respect to the other sense. Now take the simple case where the word has only 2 sense: sense a and sense b. So for a given collocation I will find out what is the probability of sense a given the collocation, and what is the probability of sense b given this collocation.

Now, question is once I found both these numbers, what will be a function that will tell me how good this collocation is. Now this collocation would be good if one of these probabilities is high another probability quite low. So collocation indicates sense a if this probability divide by this probability is high and to indicate sense b if the inverse of that is high. And this simple measure we used to come find out what collocations for a word are more important than others.

So I come this and I take a log of this and this is called a log likelihood ratio. Log of probability of sense a given the collocation I divide by probability of sense b k given collocation i. And then hire log likelihood means more evidence. So what I will do? For different collocations that I have extended for the word these can be some simple words that occur a lot with this word in different context. I will compute this log likelihood this course. And then I am I will arrange them in their decreasing order and this will give me the decision list.

(Refer Slide Time: 29:43)

### Decision List Algorithm

Training Data		Resultant Decision List	
Sense:	Training Examples (Extracted in Context)	Final decision list for plant (abbreviated)	Sense:
A	used to describe plant life ...	Log. Collocation	B
A	most distribution of plant life ...	10.12 plant growth	⇒ A
A	clean-up studies of plant life and natural ...	9.68 car (within ±k words)	⇒ B
A	too rapid growth of aquatic plant life in water ...	9.64 plant height	⇒ A
A	the most important stages of plant life ...	9.54 union (within ±k words)	⇒ B
A	establishment phase of the plant virus life cycle ...	9.51 assembly plant	⇒ B
B	computer manufacturing plant and adjacent ...	9.50 nuclear plant	⇒ B
B	discovered at a St. Louis plant manufacturing ...	9.31 flower (within ±k words)	⇒ A
B	paper manufacturing plant found that they ...	9.24 job (within ±k words)	⇒ B
B	's owners manufacturing plant in Alps ...	9.03 fruit (within ±k words)	⇒ A
B	polymer manufacturing plant at its Dow ...	9.02 plant species	⇒ A
B	company manufacturing plant in Orlando ...	...	...

Classification of a test sentence is based on the highest ranking collocation, found in the test sentences.

*plucking flowers affects plant growth.*

Pawan Goyal (IIT Kharagpur)      Word Sense Disambiguation - I      Week 8, Lecture 15

So here is an example. Suppose you have some training data and here the ambiguous word is plant, near trying to extra various collocations that can help me to find out whether the sense used is a, a is in the sense of plant life and with the sense is b in the sense of manufacturing.

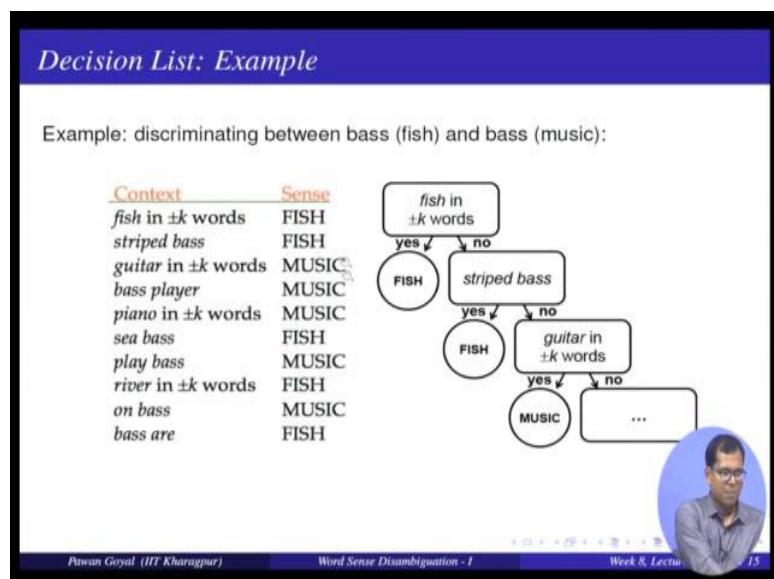
So suppose I run my algorithm and I find out that the collocation plant growth it is having a very high likelihood that is 10.12. And this for the sense a, what does that mean? Log of probability sense a given plant growth divides by probability sense b given plant growth will become 10.12. And this is the highest among all the collocations. This comes out on top n a c sense is a. Suppose, similarly the second collocation is if the word car occurs in plus minus k words around this ambiguous word plant. Then the sense would be b and this as a log likelihood of 9.68 and so on. And with each likelihood I have a sense now.

Once I have that, I can use it directly as my decision list classifier. So what I will do at runtime whenever I am given the sentence I find out if this collocation is present plant growth. If this is present senses a if not whether the word car occurs in plus minus k words around plant if so senses being. If not if plant height occurs senses a and so on. So when I get a sentence at one time like plucking flowers affects plant growth, I will take

this sentence and run it through this decision list classifier. And accordingly wherever I find a match I immediately provide the sense and I stop.

Now, one thing you have to be careful here with these numbers. So remember the formula we have a probable determine the denominator and it can also be 0 in some cases. So you might have to use some sort of smoothing you can use add one a smoothing or some other smoothing method for that. So once you do that you can come up with this decision list classifier and at runtime given a sentence you can easily compute what is a sense that should be used here.

(Refer Slide Time: 32:08)



Here is another example of decision list classifier. Like you are discriminating with me bass and base - the fish and music sense. So it can be something like that. Suppose this is how your different collocations are ordered as per your log likelihood. So we make a finishing tree if the word fish occurs in plus minus  $k$  words if yes sense is fish if no, if the collocation stripe bass occurs. If yes fish if no if the word guitar occurs is directly coming from the ordered list of log likelihood. And this you can run through a new sentence to find out the appropriate sense of the word bus.

So we saw some of the algorithms that that by using knowledge based approaches and machine learning approaches. Now in the next lecture we will also talk to talk about some other approaches that are either semi supervisor unsupervised, but in general there

are many different algorithms that you can apply on this task we only providing you very brief ideas on some of those.

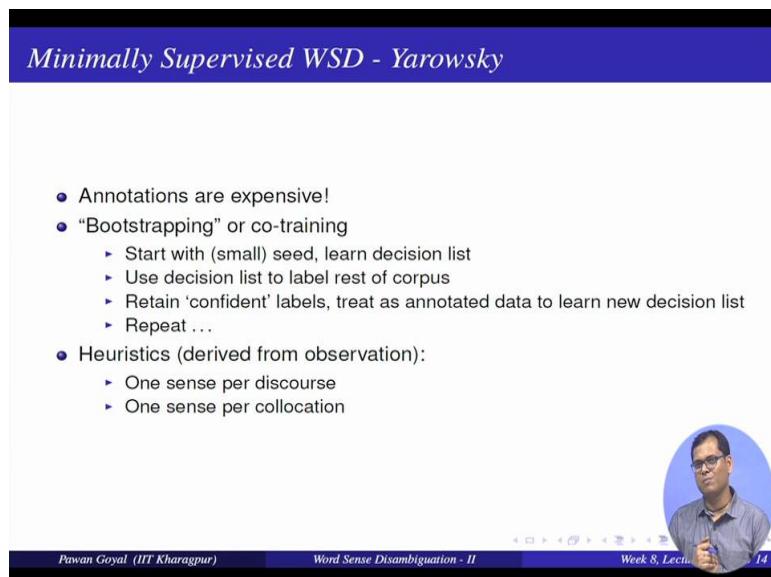
Thank you so much. So I will see you in the next lecture.

**Natural Language Processing**  
Prof. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 40**  
**Word Sense Disambiguation – II**

Welcome back for the fourth lecture of this week. So, we had already started our discussions on word sense disambiguation in the last lecture; and we talked about some approach that we use I that are either knowledge based or use some machine learning algorithm. So in this lecture, we will talk about some approaches that are either semi-supervised or unsupervised.

(Refer Slide Time: 00:41)



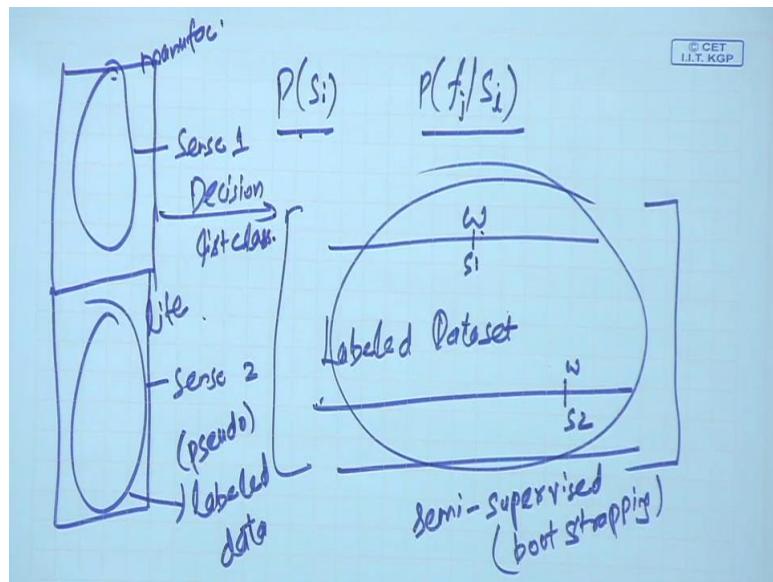
*Minimally Supervised WSD - Yarowsky*

- Annotations are expensive!
- “Bootstrapping” or co-training
  - ▶ Start with (small) seed, learn decision list
  - ▶ Use decision list to label rest of corpus
  - ▶ Retain ‘confident’ labels, treat as annotated data to learn new decision list
  - ▶ Repeat ...
- Heuristics (derived from observation):
  - ▶ One sense per discourse
  - ▶ One sense per collocation

Pawan Goyal (IIT Kharagpur) Word Sense Disambiguation - II Week 8, Lecin. 14

So, why do we need to talk about the semi-supervised, unsupervised approaches? So, in all the machine learning approaches that we can use for this particular problem or any other problem, what is one particular bottleneck let us take the Naïve Bayes algorithm. So, for applying Naïve Bayes algorithm, you want to compute the features at that are as such easy some sort of domain knowledge will be required.

(Refer Slide Time: 01:13)



But you have to find out all these numbers like what is the probability of a particular sense, what is the probability of a feature given a sense, yes, now how do you get all these values. For getting all these values, you need to have some sort of label data set. And what do I mean by labeled data set in this context, I will have various sentences where this is my ambiguous word has occurred and somebody has labeled it that this is sense 1. In this sentence the word w occurs this is sense 2 and so on. So, in some good number of sentences somebody has labeled each word as belonging to one or the other senses; and from there, I can compute all these values. And this has to be done for each individual word, and this can be really, really expensive getting all this annotations them and that is true for any generic machine learning task.

So, need to have all these annotations first and that is quite expensive that is why if you can find some sort of semi-supervised approach where starting with very few labels you can generate lot of different labels, and which may not be 100 percent accurate, but you can assume that they are mostly correct. So, that is also quite well appreciated that is you are trying to use some sort of semi-supervised approach, and that will save a lot of efforts in labeling, and also you use a lot of bootstrapping here.

So, you see one such approach for word sense disambiguation that was proposed by Yarowsky. So, this is also called minimally supervised word sense disambiguation. And what is the motivation that for using a supervised learning algorithm, I have to use annotations and

that are quite expensive. So, I use bootstrapping, so what is the idea I start with some small seed set, and I am learning my decision classifier. From the small seed set, use my decision list label, the rest of the corpus again from there find out what are the labels that are confident, modify your decision list algorithm, again run it on the corpus, find out more labels and so on. So, you try to go in some sort of a loop of starting from some seed set, labeling some sentences, extending your seed sets of our patterns or rules, again applying it getting more labels done and so on until you have somehow converging.

So, and you can use some sort of heuristics for doing this semi-automatic labeling. And two heuristics are very common in the case of words disambiguation, one is one sense per discourse and another is one sense for collocation. One sense for collocation we have already seen in the previous example of decision list classifier. What is one sense for discourse? So, idea is that suppose you are taking one particular document or one particular long context; if the same word has been repeated multiple times most probably it is used in the same sense. So, for example, you are seeing a cricket commentary or some news article about sports and you find the word bat has been used at one point and you know this category is sense one.

Now, if the same word bat occurs in some nearby places, you can probably assume that this is also being used in the same sense, and this is the idea. So, in the same discourse, if your word appears multiple times, I tag it with the same sense.

(Refer Slide Time: 05:12)

*More about heuristics*

*One Sense per Discourse*

- A word tends to preserve its meaning across all its occurrences in a given discourse

*One Sense per Collocation*

- A word tends to preserve its meaning when used in the same collocation
  - Strong for adjacent collocations
  - Weaker as the distance between the words increases

Pawan Goyal (IIT Kharagpur)      Word Sense Disambiguation - II      Week 8, Lecture 4      3 / 14

So, these are my two different heuristics one sense for discourse that is a word tends to preserve its meaning across all its occurrences in a given discourse. And one sense for collocation we have already seen that. If it is used with the same collocation, it will have one meaning, and generally it is strong for nearby collocations, and if the distance becomes higher it becomes weaker and weaker. So now, how do we use these ideas to construct a semi-supervised approach for words disambiguation?

(Refer Slide Time: 05:46)

### Yarowsky's Method

*Example*

- Disambiguating plant (industrial sense) vs. plant (living thing sense)
- Think of seed features for each sense
  - Industrial sense: co-occurring with 'manufacturing'
  - Living thing sense: co-occurring with 'life'
- Use 'one sense per collocation' to build initial decision list classifier
- Treat results (having high probability) as annotated data, train new decision list classifier, iterate



Pawan Goyal (IIT Kharagpur)      Word Sense Disambiguation - II      Week 8, Lecture 14

So, idea would be let us take the same example that is I am having the word – plant, it has two senses, one is industrial sense, another is living thing sense. So, I will start with some sort of seed labels or some seed collocation. So, suppose I say that with the first sense that is industrial. If the word plant occurs with manufacturing, it will have only the first sense. Similarly, if the word plant occurs with life, it will have the living thing sense. So, this becomes my decision list classifier I check simply if the words plant is occurring with manufacturing or life; manufacturing, I say it is sense 1; if life, sense 2.

So, what will I do, I will take my whole corpus see wherever with the word plant one of these words either manufacturing or life occurs; wherever the word manufacture occurs I say this is labeled as this is my pseudo label sentence. So, now I have some set of pseudo label sentences both with manufacturing and using manufacturing and life. So, what I would have now, from my corpus, I would have labeled some sentences with sense 1, with sense 2. So,

these are all the sentences where the words plant occurs with manufacturing and these are all the sentences where the plant occurs with life.

So, now the idea would be. So, now, I am doing going iterations. So, now I will treat them as my labeled data. Although to be very clear, we can also call it my pseudo labeled data. It is like I am only making an assumption that they are labeled with sense 1 and sense 2. Now, treat it as your label data and then apply your decision list classical algorithm to find out what are the good collocations.

Suppose, you find some three, four collocations here, so it can be different things about growth and whatever or the word car like what we saw in the previous case. So, we find some collocations here. Now, use this collocations and your algorithm to build a new decision list classifier - new classifier. Again you will use that to label new sentences. And you will have new data, you will get new collocations from there, and you keep on repeating the system.

(Refer Slide Time: 08:32)

### Yarowsky's Method: Example

Pawan Goyal (IIT Kharagpur) Word Sense Disambiguation - II Week 8, Lecture 4 5 / 14

So, let us see for this particular example. So, suppose I use that and I label some sentences with sense of living thing, some with sense of industrial plant. So, you see here everywhere life is occurring, here everywhere the word manufacturing is occurring. Now, from these labeled or by pseudo labeled data, I will try to extract some more collocations like here suppose I see that the word like animal and kingdom occur a lot with this sense, but they do not occur with this sense.

So, this becomes my collocation now. Now, use your decision list classifier algorithms previously that we have talked about to make my decision list. Now, again run it over the corpus, and then you can find suppose the word where plant and animal occurs. So, here you can again label it with sense of sense of living thing and from once you have done that you can extract some more collocations from here and you keep on doing that.

(Refer Slide Time: 09:35)

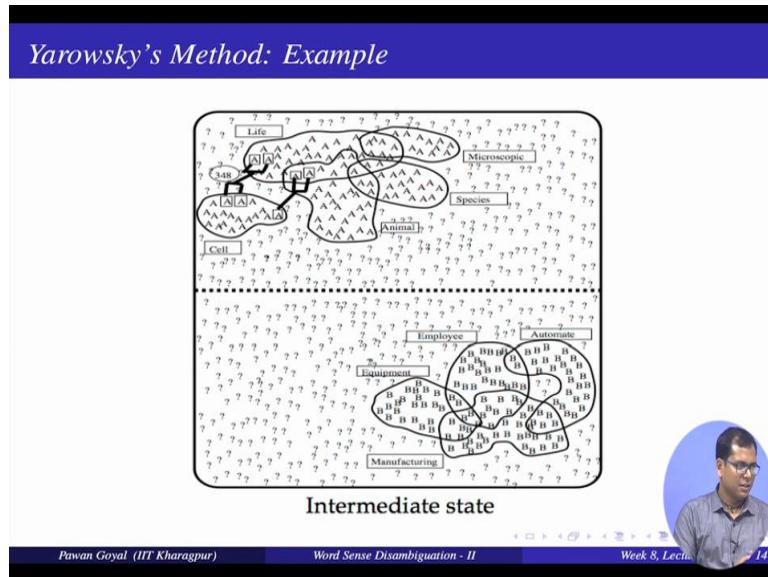
*Yarowsky's Method: Example*

Initial state after use of seed rules

Pawan Goyal (IIT Kharagpur)      Word Sense Disambiguation - II      Week 8, Lect. 14

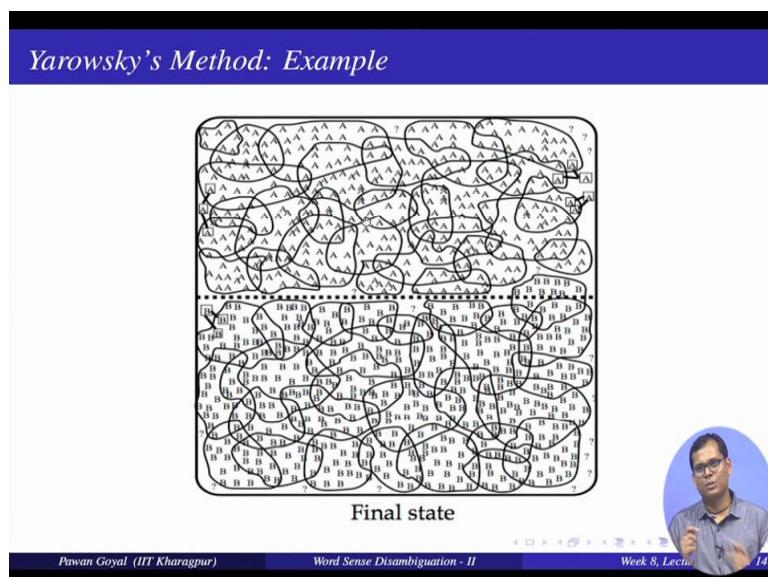
So, idea would be something like that. So, you have this whole corpus where the word plant occurs somewhere by using this initial seed set of life; and manufacturing you have label sense a and sense b. Now, what you will do, you will treat them as label data from their capture some more collocations.

(Refer Slide Time: 09:58)



So, something like this. So, you capture collocation like cell, animal, species, microscopic for the sense of life; and equipment, employee, automated, manufacturing, for the sense of industrial plant. Now, use that to label more sentences and it is a more sentences that you have labeled. So, all these content this microscopic species, animal and so on. Now, you have a larger data, again extract collocations from where build your entries and try to label the rest of the corpus, so you keep on iterating this. And ideally at some point you will have you whole corpus covered

(Refer Slide Time: 10:32)



(Refer Slide Time: 10:40)

The slide has a blue header bar with the title "Yarowsky's Method". Below the header, there are two main sections: "Termination" (in a light purple box) and "Advantages" (in a pink box). The "Termination" section contains three bullet points: "Stop when" (with sub-points "Error on training data is less than a threshold" and "No more training data is covered"), "Use final decision list for WSD". The "Advantages" section contains two bullet points: "Accuracy is about as good as a supervised algorithm" and "Bootstrapping: far less manual effort". At the bottom of the slide is a video player interface with a circular video thumbnail showing a person speaking, and text indicating the speaker is "Pawan Goyal (IIT Kharagpur)" and the topic is "Word Sense Disambiguation - II", "Week 8, Lecn. 14".

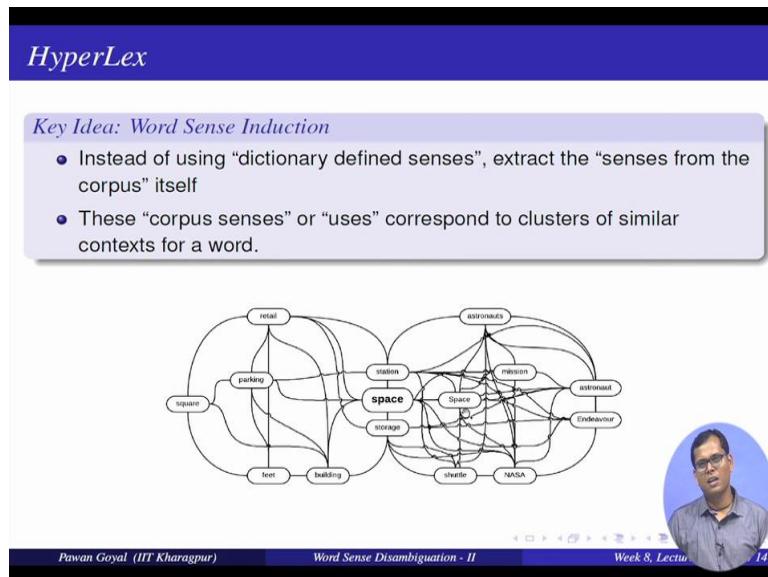
So, when do you stop, when the error on the training data is less than a threshold? So, we can have some small training data to check; how could this bootstrapping approach is performing or when no more training data is covered by the algorithm. When you see that all the examples have been covered, and whatever final decision list you get by this algorithm use that for word sense disambiguation. Now, what is the advantage of this approach? So, you can see that the advantage here are that it would have to put the efforts into manual labeling of the corpus. So, you can start with some very small seed set keep on applying this algorithm retroactively and obtain the final classifier.

Now so accuracy also turns out to be quite good, but what are the different what maybe one problem with this approach. So, one problem with this approach might be that the whole accuracy that you get by this approach depends on how good are your initial seed set. If your initial seed set is not good, then you may not go very far in this approach. So, the good thing is if you choose very good initial seeds that have lot of represent the corpus and are also very, very distinctive. You see this gives starts giving you very good precision and good recall from the very beginning. So, now this is a semi-supervised approach, and I hope the idea is clear.

Now, we will see one unsupervised approach for word sense disambiguation. By unsupervised I would mean that the senses for the words are not defined a priori, but they are sort of learned from the data. So, from the data, you try to find out what are the different

senses a word is used in, and this becomes your sense definition for each word. And this field is also called word sense induction as such. So, you are trying to induce the sense of the word by its usage in the corpus.

(Refer Slide Time: 12:52)



Now what is the basic idea? So, we will talk about the algorithm the HyperLex. So, key idea here is word sense induction that is instead of taking some sense that are defined by a dictionary, try to extract the senses from a corpus itself. So, the way the word is used in a corpus, use that for extracting senses of the word. And these corpus senses or uses will correspond to clusters of similar context for a word.

Now, let us take one example to get the intuition. So, here you are seeing the word space and there are many other words that are coming along with this space. So, the connections show here that two different words occur together above some threshold number of times in a corpus. So, as denotes that they are probably similar in the sense that they are co occurring a lot. So, now, try to look at this simple picture. So, what do you see the words like retail-parking are connected, retail-square are connected, feet and parking connected here in the left hand side; right hand side you are seeing words like astronauts, mission and NASA, shuttle all these are connected.

Now, by looking at these connections, so what is one thing that you are seeing is that you are finding out two different clusters here; one in the left hand side one in the right hand side. One is denoting one sense of space in the sense of parking space, retail space. In secondary

space as such space where astronauts will be going in and that involves NASA and all. Now, from the corpus I can find out that astronauts - NASA, and shuttle-NASA connected, retail-parking, retail-square all these are connected. But how will I find out that the word space has two senses, and these two senses correspond to some particular words.

So, if you look at the figure again, you can see that for a particular sense of the word space, the words will be highly connected to each other; for another sense again these words are highly connected to each other, but there will not be much connections between the words in this sense and words in that sense. So, that is if I construct this as a graph and use some sort of clustering algorithm to find out what are the different portions of partitions of this graph.

So, one partition will belong to one sense another partition will correspond to another sense. And this is the idea try to exploit how much that the difference for the same sense the words will co-occur a lot together, but it will not happen for the words across two senses. And there are many, many different algorithms that are developed based on this idea for word sense induction.

(Refer Slide Time: 16:07)

The slide has a blue header bar with the text "HyperLex". Below it is a light purple section titled "Detecting Root Hubs". Inside this section is a bulleted list of steps:

- Different uses of a target word form highly interconnected bundles (or high density components)
- In each high density component one of the nodes (hub) has a higher degree than the others.
- **Step 1:** Construct co-occurrence graph,  $G$ .
- **Step 2:** Arrange nodes in  $G$  in decreasing order of degree.
- **Step 3:** Select the node from  $G$  which has the highest degree. This node will be the hub of the first high density component.
- **Step 4:** Delete this hub and all its neighbors from  $G$ .
- **Step 5:** Repeat Step 3 and 4 to detect the hubs of other high density components

At the bottom of the slide, there is a navigation bar with icons for back, forward, search, and other presentation controls. The footer contains the text "Pawan Goyal (IIT Kharagpur)", "Word Sense Disambiguation - II", "Week 8, Lecture 4", and "11 / 14".

So, now we will talk about a very simple approach HyperLex that uses this idea. Now, what is this algorithm HyperLex. So, what it does is that for a given word that it ambiguous by using the data in my corpus, it tries to identify what are the main hubs. So, each hub will correspond to one sense of this word. So, it tries to identify, what are the main hubs and every other word in my co occurring graph will be connected to one or the other hubs. So, you

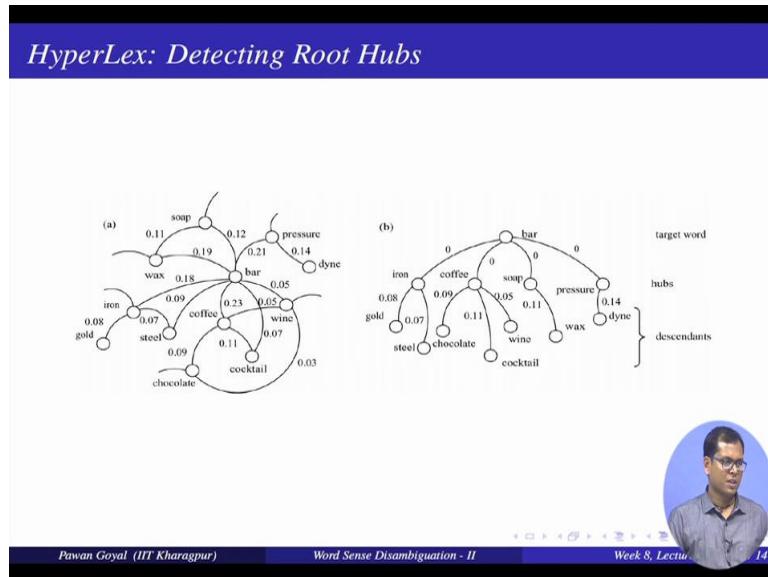
know everything is divided into these hubs. So, how do you detect these hubs and how do you connect the words to one of these hubs.

So, in HyperLex algorithm, the idea is that the different usage of the target words from highly interconnected bundles or high density components; and each component will have one of the nodes this is hub that is having a high degree than the other nodes. So, how do you start applying this algorithm? So, firstly you have to construct your co occurrence graph. Co occurrence graph we have seen already in the previous week that I find out how many times these two words occur in the corpus and I use some function of that to find out how much their association is strong how strong is their association. And I will probably retain only those words that have a very high association. So, I start by connect correct building this graph from the corpus co occurrences.

Now, so what do you do this graph is connected around this word this ambiguous word like a space in the previous example. So, first thing I will do I arrange all the nodes in this graph in the decreasing order of their degree. So, I will forget about the word space, but every other word will be connecting the decreasing order of their degree. So, what I will assume that whatever the hubs are will have high degree shape. So, take the node that is having the highest degree among all the connections and this will be the first hub. Try to find out what is the neighborhood of this hub take them to this sense cluster remove this from the graph altogether from the remaining graph find out what is the node with the highest degree make it the second hub. Find out its neighborhood make it the second sense cluster remove it, keep on doing that.

So, once I have arrange the nodes in the graph  $G$  in decreasing order of degree, now select the node from  $g$  that has the highest degree and this will be the hub of the first high density component. Now, delete this hub and all its neighbors from the graph. From the remaining graph, again repeat the step 3 and 4 to find out what are the hubs and what are their high density components.

(Refer Slide Time: 19:00)



So, let us try to see this algorithm on a simple example. So, here what do you have, you have the word bar that is the central word; and I want to detect what are its different senses. So, what we have done we have first try to construct the co occurrence graph. So, this is what you are seeing is a co occurrence graph that tells what is the strength of association between two words. One thing here, I will talk about what is association measure. So, according to this association measure, you will take a threshold and choose only those edges that are above or below the threshold depending on how you are defining your threshold. In this particular case, so the number denotes what is the distance between two words. So, you are only retaining those words whose distance is below a threshold. So, say your threshold is 0.25, so we written only those connections that are below 0.25. And distance can be captured by something that is inversely proportional to the association.

(Refer Slide Time: 20:18)

A handwritten mathematical expression on a grid background. The formula is  $1 - \max(P(w_i|w_j), P(w_j|w_i))$ . A bracket under the max function is labeled "Distance". To the right of the formula, there are two circles containing the letters  $w_i$  and  $w_j$ . In the top right corner of the slide, there is a small logo with the text "©CET I.I.T. KGP".

$$1 - \max(P(w_i|w_j), P(w_j|w_i))$$

So, in this case it is 1 minus probability of max of probability. So, it is 1 minus max of probability  $w_i$  given  $w_j$ , and probability  $w_j$  given  $w_i$ . So, we take the max of that to find out what is the association would be  $w_i, w_j$ . And this will now capture the distance, because this corresponds to the similarity how similar they are, their condition probability. If I take 1 minus max of that that is how much they are different what is the distance. So, once we have done that for all the words, all the pair of words, I will take only those that are having distance below a threshold that means, they are quite clear and that is captured in the left hand side of the figure.

Now, once you have done that now you apply your algorithm that is finding out what are different hubs and taking their different neighborhoods. So, first I will arrange all the words in decreasing order of the degree. Now, what are the nodes here that are having the highest degree? So, you find the words like iron, iron has a degree of 1, 2, 3, and 4; coffee has degree of 1, 2, 3 and 4; wine also has a degree of 1, 2, 3 and 4. So, now there are multiple words that are having the highest degree. So, we will choose some preference may be either the lexical graphical order on some or some ordering on the, or some travels or ordering on the graph. And suppose you say from I will choose the one from the left, and you choose the word like iron here as your first hub.

So, what is the next step make iron as your first hub and then take all the neighbors of iron and put them with this hub. So, we are taking the two neighbors gold and steel and putting them with this hub here. And that becomes you first HyperLex components. Now how do you find the next hub, you remove this hub and all it is neighbors. So, I remove iron, gold and steel from the graph. Again find out what is the node with the highest degree? You will find the word like coffee here. So, take coffee of the second hub take its neighbors, so it will be chocolate, cocktail and wine. So, these become its neighbors, this becomes my second hope hub and all the component with that.

Remove that from the graph now find out the next hub. So, this can be the word soap that is having a degree 3, you take the words soap and wax as your third hub and its component. And then finally, you will have the word like pressure and dyne that is your four hub. So, this is my hub and these are different descendants of the hub. So, from your corpus, you are focusing one word k bar, and you are trying to construct the co occurrence matrix finding out the association between various words, building out this graph, from there you are detecting what are your various hubs and the descendants. And this becomes your instruction; this is your target word different hubs and their descendants. And this is what you have obtained in a completely unsupervised manner, because nobody told you how many senses the word bar would have or what are the different senses of bar you found it automatically by using the corpus data.

(Refer Slide Time: 23:54)

### Delineating Components

- Attach each node to the root hub closest to it.
- The distance between two nodes is measured as the smallest sum of weights of the edges on the paths linking them.

*Computing distance between two nodes  $w_i$  and  $w_j$*

$$w_{ij} = 1 - \max\{P(w_i|w_j), P(w_j|w_i)\}$$

where  $P(w_i|w_j) = \frac{\text{freq}_{ij}}{\text{freq}_j}$

$$w_{ij} = 1 - \max\{P(w_i|w_j), P(w_j|w_i)\}$$

where  $P(w_i|w_j) = freq_{ij}/freq_j$

So, for all other words other than hub, you attach them to the root hub that is closest to them. And how do you find what is the closest hub you can take the distance between that node and the different route hubs. And this distance is simply the summation over the path length. What is the path length you just keep on adding the path length that is the distance between any node and the root hubs, attach each node to one of the root hubs only. And this is something that I talked about that how do we compute the distance between two nodes in my original co occurrence graph I take this 1 minus max probability of w i given w j and probability of w j given w i.

So, we can see that how do we start from my corpus and construct different senses of a word, so y hub and descendants. Now at run time, so we are talking about this problem what is in the disambiguation. So, at run time, I am given a sentence where this word is provided; and I want to find out what is it is appropriate sense that is used. Among all the possible senses I have fine formed by this algorithm. So, what is the approach?

(Refer Slide Time: 25:14)

## Disambiguation

- Let  $W = (w_1, w_2, \dots, w_i, \dots, w_n)$  be a context in which  $w_i$  is an instance of our target word.
- Let  $w_i$  has  $k$  hubs in its minimum spanning tree
- A score vector  $s$  is associated with each  $w_j \in W(j \neq i)$ , such that  $s_k$  represents the contribution of the  $k$ th hub as:

$$s_k = \frac{1}{1 + d(h_k, w_j)} \text{ if } h_k \text{ is an ancestor of } w_j$$

$$s_i = 0 \text{ otherwise.}$$

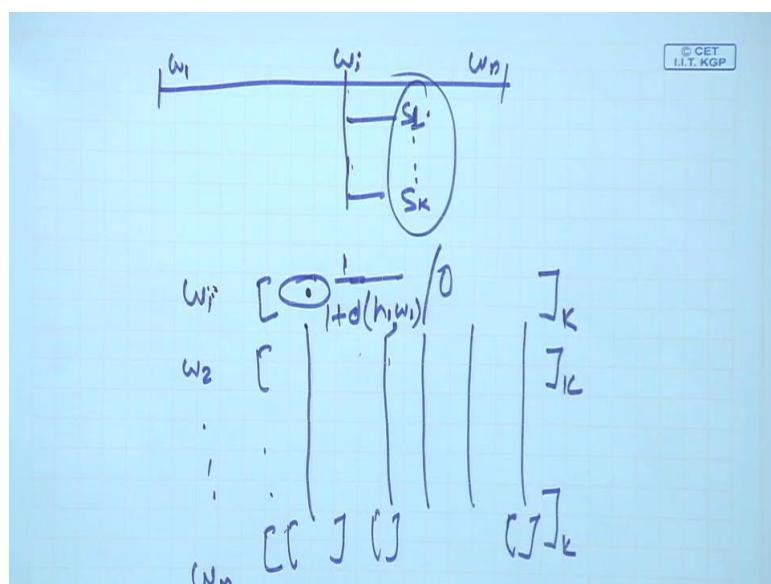
- All score vectors associated with all  $w_j \in W(j \neq i)$  are summed up
- The hub which receives the maximum score is chosen as the most appropriate sense

Pawan Goyal (IIT Kharagpur) | Word Sense Disambiguation - II | Week 8, Lecture 4 | 14 / 14

$$s_k = 1/(1 + d(h_k, w_j))$$

So, let us say I have this context or the sentence where this word  $w_i$  occurs and  $w_i$  is my target word that has multiple senses. So, suppose they are it has  $k$  senses,  $k$  hubs that are  $k$  senses. So, what do we do, we associate a score vector  $s$  with each word in the context such that  $s_k$  denotes what is the contribution of the, what is the contribution it will have to the  $k$ th hub. And this is simply taken by 1 divide by 1 plus distance between the hub and the word. If the hub is an ancestor of the word  $w$  otherwise it is 0. And you do that for all the words in my context and sum those over. So, find out a simple vector that tells me which sense has how much score whichever have how much score and whichever has hub has the highest score I will choose that.

(Refer Slide Time: 26:28)



So, to tell that again, so what we will do, so we will have a sentence  $w_1$  to  $w_n$  and I have this word  $w_i$  that has  $s_1$  to  $s_k$ ,  $k$  hubs. Now, I want to find out which of these  $k$  senses is used in this particular example, particular sentence. So, what will I do for each word  $w_1$ ,  $w_2$  up to  $w_n$ , I will construct this score vector. So, score vector has that many dimensions as the number of hubs, so it has  $k$  dimension. So, I will construct this vector for each of the words.

So, now what are the entries in this vector? This entity tells me how much contribution this words will make to the first hub. And this contribution would be if the particular hub  $h$  of the sense one is an ancestor or this word  $w_1$ , it will be 1 divided by 1 plus distance of  $h_1$  and  $w_1$ . So, that is if the distance is high this score will be low if the distance is smaller this will give a high score.

On the other hand, if this word, if the hub is not ancestor, I will put a score of 0; like that I will put all the different values here. And finally, once I have all the values, I will add all these. So, for each hub, I will add all the possible scores; and I will take a final vector that is how much contributions all the words together are making for hub 1, hub 2 and hub k. And I take the one that is having the maximum score among all these and that becomes my winner sense. So, this is the overall idea of this approach. So, what we have done here we did not start with any distance defined sense, we used a corpus and defined our own senses by seeing what are the different components that occurred together. So, idea is that for a given sense of a word different words would have a high co occurrence they will make some sort of cluster. So, identify different cluster size different senses.

Now, once you have these senses if you want to use them for word sense disambiguation at run time whenever the word is used, find out from the context words which of thus different hubs they are closer to. So, whichever hub gets the high score becomes your disambiguate sense, so these are some different ideas for approaching this problem words and disambiguation. Although I said earlier, there are many, many different approaches that have been proposed.

So, in the next lecture, we will briefly talk about an idea of word sense discovery that is how do you find out if the word has got a new sense in the corpus. This is a relatively new field, and we will see how the ideas that we have developed in these lectures to how do we use them to find out if a word has got new sense in the corpus.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

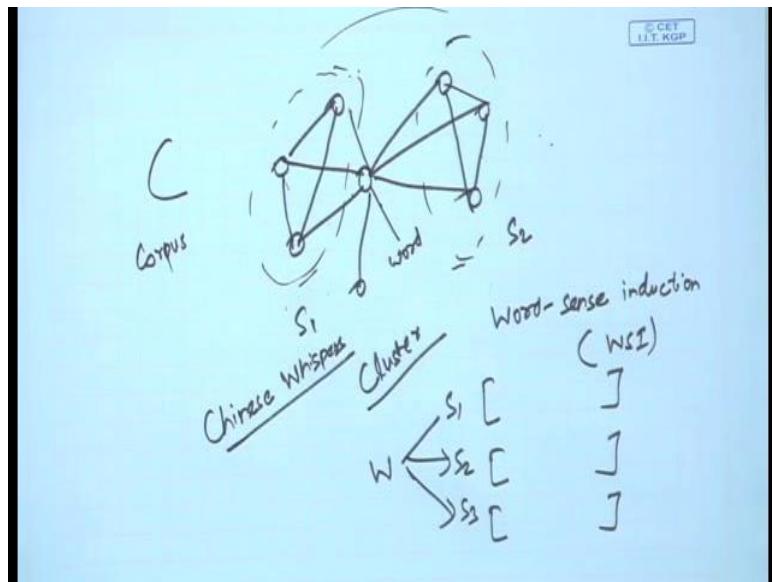
**Lecture - 41**  
**Novel Word Sense Detection**

Hello everyone, so welcome to this final lecture of week-8. So, in this week, we have talked about a different method of semantics that is lexical semantics. And we saw that how we can use connection between words to find out semantics. And last two lectures, we talked about a very classical problem of word senses, disambiguation that is even a word if it has multiple senses, how do you find out in a given context what particular senses has been used. And this is a very generic kind of problem, and you can use a lot of different methods for solving that. And we saw that how you can use simple knowledge base approaches, by using the dictionary definitions and you can use some machine learning methods, you can use some bootstrapping based methods, and also unsupervised methods.

There we also talked about a page rank based algorithm if you want to jointly find out the senses for each word. We did not describe that algorithm fully. So, if you want to know in that algorithm details, so you might have to wait till we cover the summarization topic. In summarization, I will talk about page-rank in detail, but as such the generic idea you can also get from what we discussed and also you can look it up.

So, this lecture we are taking a new research problem that is coming from word sense disambiguation idea itself, and also from whatever we discussed in the towards the end of in the last lecture. So, towards the end of the last lecture, we were talking about learning the word senses in an unsupervised manner that is now I am not talking about senses defined in a dictionary. So, I am saying I have a corpus, I know how the word has been used multiple times, can I use the usage of the word to find out what are its senses and what was the basic idea.

(Refer Slide Time: 02:23)



So, basic idea was that if I know that I have a corpus see my corpus, and I can find out which words co-occurred other words. So, like that I can construct some model I know what is co-occurred with what are others. So, idea was it suppose I am constructing this network, if a word has two senses, what will happen? The words that correspond to it is, there is suppose my center word. So, words that I connect corresponding to one sense will be connected together; words that are in second sense will be connected together. This is sense one; this is sense two. So, if I am building the whole co occurrence graph, if I center it around the particular word and try to cluster it, I can find out it is different senses. It is a very generic idea and this I call as word-sense induction also known as from WSI.

There are many ways of doing that because they are many ways you can do this clustering you can apply any graph based clustering method. So, for example, Chinese Whispers is a very popular algorithm and you can use any other methods also. So, idea would be now when you do that for each word, you will get some senses  $S_1, S_2, S_3$  and so on. And these might correspond to say the words, these are the words, it can be different words they can have weights and so on. Now, so this is you have give a corpus, you can find out what are the word senses, but in this lecture, we will talking about the problem that can you find out if a word has got a new sense or not in some recent time, has it got a new sense. Now, does that happen? So, over the time, we keep on using the same words new, new and senses. So, and with the social media this is becoming a lot more common that you are using the same word in some new senses that it was never used before.

(Refer Slide Time: 04:40)

The screenshot displays a presentation slide with a blue header bar containing the title 'Tracking Sense Changes'. Below the header, there are two main sections: 'Classical sense' and 'Novel sense'.  
**Classical sense:** This section shows the Merriam-Webster dictionary definition for 'sick' as an adjective. The definition includes three meanings:

- : affected with a disease or illness
- : of or relating to people who are ill
- : very annoyed or bored by something because you have had too much of it

**Novel sense:** This section shows a tweet from Niall Horan (@NiallOfficial) posted on April 24, 2014. The tweet reads: "Listening to Paulo nutini's new record! It's sick!" The tweet has been favorited 85,144 times and has 47,293 retweets. The tweet is timestamped at 11:50 PM - 24 Apr 2014. At the bottom of the slide, there is a footer with the URL <http://www.merriam-webster.com/>, the text 'Novel Word Sense Detection', and navigation links for 'Week 8, Lecture 5' and '2 / 5'.

So, let us see one example. So, I have the word sick right; sick is a very common and popular word and you will say what is the meaning of sick something to be related to some disease or illness. So, this is the dictionary definition, so affected with a disease or illness, of or relating to people who are ill, and very annoyed or bored by something because you have had too much of it. So, I am sick of that.

So, now this is one common sense of sick. Now, do you think sick has got any new sense in recent times, and if you think about it, sick has got a very new sense and that is completely opposite of this particular sense. So, what is that sense? So, let us look at this tweet. So, listening to Paulo Nutini's new record, it is sick. So, now, what is the meaning of sick here of the same word sick, it is not boring. So, this sick means something that is very, very cool. So, this sick has got a very new meaning from whatever we were seeing earlier something that disease illness to something that is very cool.

Now, so what is keep on getting these new meanings in the corpus, so the way people are using that. Now, the problem is suppose my dictionary or my lexicon like word net is not getting updated regularly. So, if this word sick has got this new sense, and it is being used in this new sense in the corpus, I will never be able to match it to any of the sense in the word net, because word net has not recorded the sense at all. So, (Refer Time: 06:19) does not know probably sick has the sense also and this is happening for many other words.

So, now what he is done in this field of novel versus rejection, from the corpus and the way the words are being used can I detect whether the word has got new sense; and if I can detect it I can populate different lexicons and different and also my word net version I can update using these definitions. So, how do we find out the word has got new sense, now again, so this is a new area, but there have been some different sort of methods and works, I will not go into details of any of those works. But what I will do? I will try to give you a basic idea that if you understand word sense induction how can you use that to find out new word sense or novel word sense.

(Refer Slide Time: 07:10)

### Comparing sense clusters

- If a word undergoes sense change, this can be detected by comparing the sense clusters obtained from two different time periods

The slide illustrates the concept of word sense change through the example of the word "compiler". It shows two distinct sense clusters for the same word at different times.

**1909-1953 Sense Cluster:**

- reporter/NN, auditor/NN, listener/NN, scribe/NN, translator/NN, writer/NN, reader/NN, editor/NN, author/NN, orator/NN, commentator/NN, composer/NN, biographer/NN, novelist/NN, ...
- compiler/NN
- scientist/NN, compuator/NN, philosopher/NN, publisher/NN, preacher/NN, transcriber/NN, thinker/NN, teller/NN, statesman/NN, musician/NN, juror/NN, essayist/NN, interpreter/NN, observer/NN, auditor/NN, experimenter/NN, artist/NN, dramatist/NN, ...

**2002-2005 Sense Cluster:**

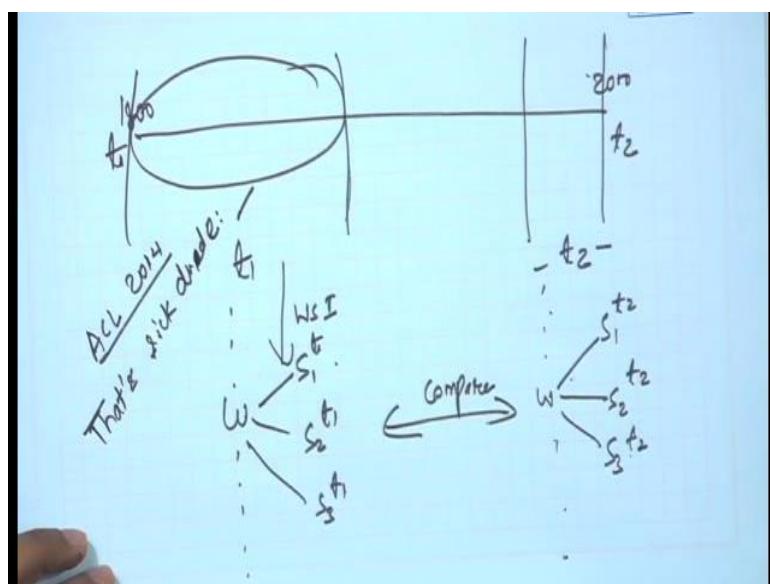
- translator/NN, editor/NN, listener/NN, reader/NN, commentator/NN, author/NN, observer/NN, interpreter/NN, writer/NN, scribe/NN, redactor/NN, viewer/NN, ...
- compiler/NN
- preprocessor/NN, driver/NN, handler/NN, hardware/NN, software/NN, loader/NN, kernel/NN, driver/NN, binary/NN, assembler/NN, scheduler/NN, debugger/NN, browsers/NN, processors/NN, parser/NN, subsystem/NN, ...

A circular portrait of the speaker, Pawan Goyal, is shown in the bottom right corner, with the text "Pawan Goyal (IIT Kharagpur)" and "Novel Word Sense Detection" below it.

So, let us have a basic idea. The basic idea is that I can try to compare the sense clusters, so that is if your word is undergoing sense change it can be detected by comparing the sense clusters obtained from two different time periods, something like this. So, you take the word compiler, suppose in 1909-1953, so we took the data and we did the words sense induction; and from the induction we found out that the word compiler has these two senses. So, one sense is in the sense of reporter, auditor, listener etcetera; second is on scientists, composer, philosopher, publisher, preacher, so there are two different senses. So, what it means is that these words come together to form one sense and these words come together to form another sense in 1909 - 1953. Compiler was merely some sort of person who wish to compile.

Now if I look at this word in recent time like 2002 – 2005, compiler has got the sense in the sense of programming language compilers, this sense was not available not there in 1909 - 1953. So, can I automatically detect that the word has got a new sense from the data itself. So, what I will do, again into (Refer Time: 08:28) I will do the word sense induction. So, suppose I do induction and I find these two senses. So, one is again the same sense translator, editor, listener, reader, commentator that is similar to what we had earlier, but you see a new sense also coming up preprocessor, driver, handler, hardware, software, loader, kernel, dbms. And you can immediately see by looking at these words in the computer sense; this sense was not available earlier so, now this is interesting observation that if I simply do word sense induction over the new time period, I find a sense cluster that was not available before and this gives me a generic framework.

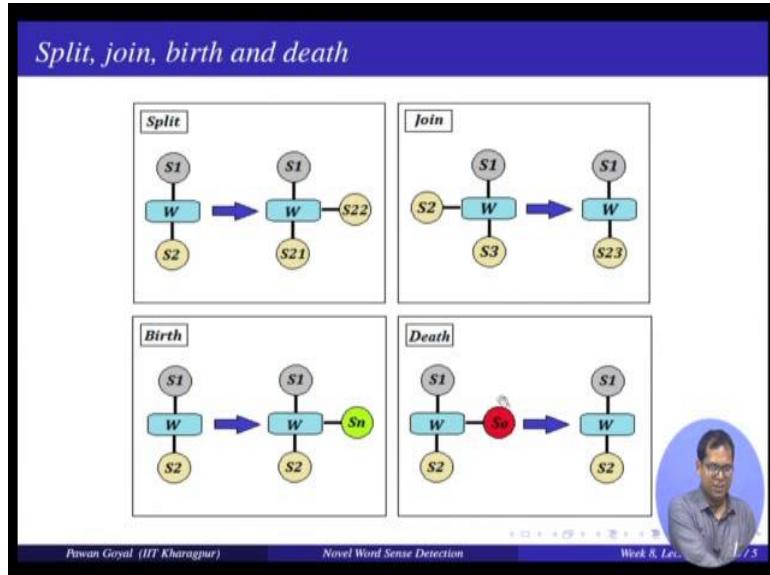
(Refer Slide Time: 09:12)



So, what I will do I will take my data that is over several time. So, it can be say 1800 to 2010 or whatever. So, it is starting from some  $t_1$  to  $t_2$ . I want to find out what words have undergone sense change. So, what I will do, I will try to take some slice of this data call it time point  $t_1$ ; take another slice later on time point  $t_2$ . So, now I will do; I will compute my co occurrence and whatever way I want to compute my distributional thesaurus then I will do word sense induction. So, for each word, I will find out  $s_1$  in time  $t_1$ ,  $s_2$  in time  $t_1$ ,  $s_3$  in time  $t_1$  and so on and that I will do for all the words. Similarly, I will do the same thing at time  $t_2$ ; so I am at  $W$ , and I will say  $s_1$  at time  $t_2$ ,  $s_2$  at time  $t_2$ ,  $s_3$  at time  $t_2$  and so on. Now, I have got the sense cluster at time  $t_1$  and at time  $t_2$ . Now, I will try to compare these

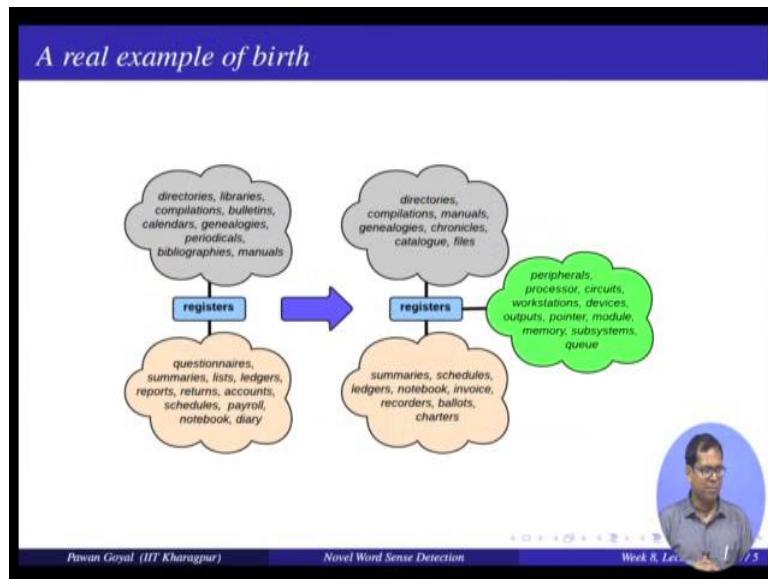
to find out if the word has got a new sense cluster that was not available in the previous time point. And the second do for all the words and by doing that I can find out which words have got a new sense in the newer time period.

(Refer Slide Time: 10:42)



So, in general I can define various different sort of sense change. For example, it might happen that earlier a word has a sense, it has got to split into multiple senses or it can be a join - two different senses joined together to form the same sense, they may not be so common. But what is common is something like a birth, that the word had initially two senses **s 1** and **s 2**, now in the new time period, it has got a new sense **s n**, this is very, very common. And then they might be death of a sense also that earlier it had a sense of **s 0**, now in the new time period I cannot sense find out the sense. So, all these you can find out just by comparing these sense clusters.

(Refer Slide Time: 11:29)



So, here is another example that we observed from the data. So, this is from the work that we did in 2014. So, this was in ACL 2014, if you want to have a look. So, the title was “That’s sick dude” and there was some other subtitle, so this was in ACL 2014. So, we took the data from Google anagrams, and we took the whole data from starting from somewhere around 1600 to 2008 divided into eight different time points then so that they were divided such that in each time point you have roughly the same amount of data.

So, as you go over recent years even like three years and four years for making complete time duration; in earlier it might be 100 years together because the data was not too much. And then we took different time points, constructed distribution thesaurus and then you Chinese (Refer Time: 12:34) algorithm to find out sense clusters and then complain the clusters. So, what are the words that are getting some new sense? So, here was an one example. So, we found that the word registers earlier had these senses, and you can see these are like dictionaries, directories, libraries, compilations, bulletins these are registers as you like paper registers and so on so that you have even now; similarly, here notebook, diary, returns, accounts. So, in the new time period, we found that these two senses are there directories, compilations, manuals, summaries, schedules, ledgers, notebook, but very new senses come up. And can you think of the sense what is the sense peripherals, processor, circuits, workstations devices. So, register has got the sense of if the sense of computing in a hardware that.

So, this new sense we could detect simply by using the corpus that these all words for being used with resistance and they were having similar co occurrence such as the word to resistance. So, we found this is a new sense cluster that is coming up and like that we did in many other senses. So, this is as I was saying this is a new research field and some works have come up in this field this is one such work, but I hope the basic ideas care that if you want to detect new word sense what you need to do, so that ends this week of lexical semantics.

So, next week again we will continue with the topic of semantics. So, we have started with distributions of semantics - one idea, then we talked about lexical semantics another idea of semantics. Now, we will see how we can capture semantics using topics, can we find out what are the topics that are there in my data use that for some semantics and that is why we will discuss topic models in detail. And there are again very, very popular tools in NLP.

So, thank you, I will see you in the next week.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 42**  
**Topic Models: Introduction**

So everyone welcome back to week 9 of this course. So we have been talking about semantics in the last 2 weeks. And we discussed 2 different approaches of semantics: one was using lexical semantics and otherwise using distribution semantics. And we saw how used the corpus or from a lexicon. You can extract semantics you can capture the meanings of words in some sense. That is whether 2 words are similar to each other and many other aspects.

So in this week we will discuss another very interesting way of capturing semantics by using topic models. And they have been very popular tools in NLP for whenever you have to talk about what are the concepts that are there in a particular document in a particular corpus and many other questions that that is around that are around the semantics. So let us see what are these topic models.

(Refer Slide Time: 01:09)

**Why Topic Modeling?**

**Information Overload**

As more information becomes available, it becomes more difficult to find and discover what we need.

**Main Tools: Search and Links**

- We type keywords into a search engine and find a set of related documents
- We look at these documents and possibly navigate to other documents

Pawan Goyal (IIT Kharagpur)      Topic Models: Introduction      Week 9, Lecture 1      2 / 11

Firstly, let us start the brief motivation that why do we actually need topic models as such. So when we talk about the data, so the text data in general on under that. So there is a huge amount of data available right in form of whatever form. You can think of there

are lot of data in terms of news articles scientific articles and blog social media and everywhere.

So you can think of it as some sort of information overload. And you might be interested to capture only certain aspect of the data. So we want capture certain semantics. So what is an easy way in which you can some sort of organize this data and then this can help you to search or browse through the data more much more effectively. So right now what about main tools that we use for doing this search of different information. Either we type some query in our search engine right some human. So free text query we type and we get some results, and we try to browse the results and sometimes results are not relevant or even if they are relevant we go to the pages to which the link to and keep on browsing. So that is some sort of generic behavior of how we try to explore this sort of information.

But there is no easy interface where you can say I want to understand these topics only I want to go too deep into this topic I want to go related topics and all that. That is not available with this search and link kind of behavior. So the topic modeling gives you an alternative method of going through this huge amount of information by some sort of searching based on themes criteria.

(Refer Slide Time: 03:00)

*Why Topic Modeling?*

*Search Based-on themes*

- Imagine searching and exploring documents based on themes that run through them.
- We might “zoom-in” or “zoom-out” to find specific or broader themes
- We might look at how themes change through time, how they are connected to each other
- Find the theme first and then examine the documents pertaining to that theme

Pawan Goyal (IIT Kharagpur)      Topic Models: Introduction      Week 9, Lecture 1      3 / 11

So you can think of it as if you are having a set of documents and you know what are the themes that are running through this corpus. So you know this documentary about certain

themes related to politics this document is about certain theme related to dramatic and so on different concepts.

Now, you are going to a popular thing and then you can either zoom in or zoom out. So that is you want to go inside the theme to very sub themes or you want to go out to a broader theme. So can this be fascinated by it by some sort of modeling? So we might also want to look at how the themes in a corpus are changing over time. How they are evolving over time this happens a lot in for example, scientific article a particular research in in physics might be starting with certain concepts and over that period of time it starting with it is now having new concepts, so by various discoveries. So can we have discovered that what are the themes setup already over time. And you can also talk about this behavior where you have you would select the theme first and then you try to examine the documents that talk about that theme.

(Refer Slide Time: 04:14)

The slide has a dark blue header bar with the title 'Why Topic Modeling?' in white. Below the header is a large white area containing a purple rounded rectangle. Inside this rectangle, the word 'Topic Modeling' is written in a light blue font. Below it, a paragraph of text is followed by a bulleted list. At the bottom of the slide, there is a circular profile picture of a man, a navigation bar with icons, and a footer bar with text and logos.

**Topic Modeling**

Provides methods for automatically organizing, understanding, searching and summarizing large electronic archives without any prior annotation or labeling

- Discover the hidden themes that pervade the collection
- Annotate the documents according to those themes
- Use annotations to organize, summarize, and search the texts

Pawan Goyal (IIT Kharagpur)      Topic Models: Introduction      Week 9, Lecture 1

So topic modeling, it is a method that gives you this facility by which you can organize all these collections by the themes that are occurring in in those documents and you can then understand search and do summarization and many other applications, without and this is important that for doing all that you do not have to give any prior manual efforts in in labeling the data. So you do not have to tell that this document concerns this topic or this document concerns that topic you do not have to do any labeling. So the interesting aspect is that you give it a corpus an in an uncivilized manner it learns what is

the overall structure of different topics and these document concerns which topics and so on.

So this is very, this is some sort of very interesting aspect that made topic modeling very popular that, you no need to have any prior annotation as such. Although there are some variations we will also talk about, where you can give an annotation to get some different sort of topic modeling, but overall in the generic picture you do not have to give any sort of annotations. So it learns on its own from the huge amount of data.

So what you do here. So you discover what are the hidden themes that are pervading to the collection and the topic model itself annotates the document as for these themes. So it will tell you these are the overall themes this document contains these themes and So on. And once you have these annotations that are given with the topic model you can use that to for validation task like organizing the collection summarizing collection searching when the user query comes by using each annotation. So many different applications, we will cover some of those in this week and, yes we can there is a huge amount of literature around topic models. So you can also feel free to read about it.

(Refer Slide Time: 06:08)

Applications: Discover Topics from a corpus			
human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

So let us talk about one of the applications before we go into the modeling in detail. So we talk we are we are consistently talking about discovering topics from a corpus. That is one of the most important application of topic models. So something like that. So you have a large corpus and can you discover that there are many topics and topic look

like this. By topic I mean a set of words that occur a lot in that topic. So let us see one topic. Human genomic dna genetic genes sequence gene molecular sequencing map information.

So topic model we will try to get you, this is one sort of topics. It will not give you a label. So although you can see this looks like a genetics topic of genetics it will not give you a label, it will tell you that this is there is a topic in this corpus there is a theme, and in this theme each words are more having higher probability. So these are the more important words in this theme. And then it will tell there is general theme that is going through this corpus, something like this evolution, evolutionary species organization life origin biology. And then you see a set of terms. These are having a high probability in that theme there is no labels that topic model is giving, but you can see it might be evolutionary biology. Then similarly another theme, disease, host bacteria, disease so on and 4 theme computer models information data computers and so on.

So the topic models will tell you in this collection you have these 4 themes in addition to some other themes. So that is your predefined number that you will need to give to the models. So you can say I want 50 different themes in this corpus, or 20 different themes in this corpus. Yes, although there are again variations where you might not have to specify these number a prior we will briefly talk about those also.

But for now let us say we tell the model we need that money themes. So try to discover these are the important themes that I am seeing with a pervading through this corpus. And these are some examples that we are seeing here that are coming from a real corpus by doing the LDS, applying the LDA model.

Now, once you have these themes then the topic model will also tell you this document is about having only these 2 themes out of these 50. This document is having these 5 themes out of these 50, in what proportions and so on. So like that you can now think about organizing huge your whole collection and also some sort of a summarization or searching through this.

(Refer Slide Time: 08:38)

*Intuition: Documents exhibit multiple topics*

### Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK—How many **genes** does an **organism** need to **survive**? Last week at the genome meeting here,<sup>1</sup> two genomic researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known **genomes**, concluded that today's **organisms** can be sustained with just 250 genes, and that the earliest life forms required a mere 128 **genes**. The other team, using a simple **parasite** and estimated that for this organism, 820 genes are plenty to do the job—but that anything short of 120 wouldn't be enough.

Although the numbers don't match precisely, these **predictions** "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Stig Andreason of Uppsala University in Sweden, who arrived at the 820 number. But coming up with a consensus answer may be more than just a **genetic numbers game**, particularly as more and more **genomes** are completely mapped and sequenced. "It may be a way of organizing any newly sequenced **genomes**," explains Arashad Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

"organism and its parasite-specific genes < 120 genes." **Adapted from NCBI**

**Neoplasma** genome: 1700 genes  
Genes shared by both organisms: 123 genes  
**Mycoplasma** genome: 448 genes

**Stripping down:** Computer analysis yields an estimate of the minimum modern and ancient genomes.

\* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

This article is about using data analysis to determine the number of genes an organism needs to survive

Pawan Goyal (IIT Kharagpur) Topic Models: Introduction Week 9, Lecture 1 6 / 11

So why is this intuition interesting? So when you look at a particular document in general you will see that yes there are actually some particular themes that are that are pervading this whole document. So that is you have somehow thought of making a document that concerns these themes only. So here is an article from science magazine 1996 Seeking Lifes Bare Genetic Necessities and so if you read this article, this is about using some sort of computational data analysis to determine the number of genes and organisms that needs to survive. So how many genes and organism needs to survive.

So this topic this article will blend some of the topics that are important to convey this message. And these words are denoted in various colors here. So we are seeing some words with pink some words with yellow somewhere to blue and so on. Let us read the words in in pink. So they are organism, survive, life, organisms, yellow, genes genomes, genes genetic, sequenced genome and blue will be computer productions numbers, computational computer analysis and so on. So you can clearly see 3 different themes one is about genetics another is about the evolution another is about this computational data analysis.

So this article is blending these 3 themes together. It is can be capture that in an uncivilized manner without someone has to manually annotate this this is the themes in this document.

(Refer Slide Time: 10:24)

*Intuition: Documents exhibit multiple topics*

### Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genome meeting here,<sup>2</sup> two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other team, using a simple parasite and estimated that for this organism, 820 genes are plenty to do the job—but that anything short of 120 wouldn't be enough.

Although the numbers don't match precisely, these predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Stig Andersson of Uppsala University in Sweden, who arrived at the 820 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arash Mehtaian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

Neurospora genome: 1700 genes  
Genes common to all: 123 genes  
Mycoplasma genome: 448 genes

Genes unique to each: 271 genes → 123 genes

Minimal gene set: 128 genes

Accessory genes: 4 genes

Human and mouse genomes: >172 genes

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

\* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

Highlighted words: 'blue': data analysis, 'pink': evolutionary biology, 'yellow': genetics

Pawan Goyal (IIT Kharagpur) Topic Models: Introduction Week 9, Lecture 1 6 / 11

So if I yes you see blue is data analysis pink is evolutionary biology and yellow is genetics. So this article is blending these 3 topics in different proportions. And that is the motivations that is what is the hypothesis that we are having about your corpus. In the corpus there will be various documents, there will be a there will be some big number of themes that are going through this corpus, but when you look at a particular document it will have only a subset of these themes, in some proportions can we automatically capture those by using some modeling.

(Refer Slide Time: 11:02)

*Intuition: Documents exhibit multiple topics*

### Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genome meeting here,<sup>2</sup> two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 820 genes are plenty to do the job—but that anything short of 120 wouldn't be enough.

Although the numbers don't match precisely, these predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Stig Andersson of Uppsala University in Sweden, who arrived at the 820 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arash Mehtaian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

Neurospora genome: 1700 genes  
Genes common to all: 123 genes  
Mycoplasma genome: 448 genes

Genes unique to each: 271 genes → 123 genes

Minimal gene set: 128 genes

Accessory genes: 4 genes

Human and mouse genomes: >172 genes

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

\* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

Knowing that this article blends those topics would help situate it in a collection of scientific articles

Pawan Goyal (IIT Kharagpur) Topic Models: Introduction Week 9, Lecture 1 6 / 11

So once you know that this article blends this topic together you can situate that in an in a collection of scientific articles. You can say where this where this situates what are the articles it is similar to and so on.

(Refer Slide Time: 11:17)

*Topic Model: Basic Idea*

A generative statistical model that captures this intuition.

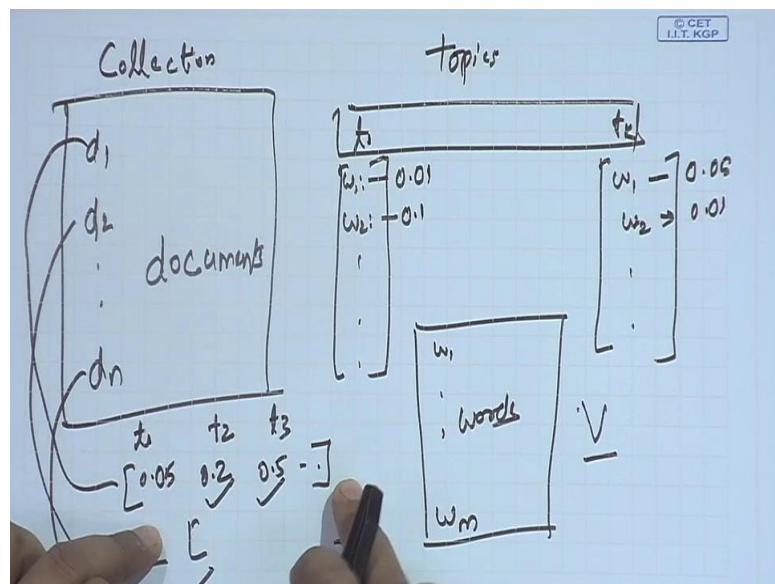
*Generative Model*

Documents are mixture of topics, where a topic is a probability distribution over words.

Pawan Goyal (IIT Kharagpur)      Topic Models: Introduction      Week 9, Lecture 1      7 / 11

So what is the basic idea? You are not going to the mathematical details that we will cover in the next week, but let us see the intuition. So this is the important idea. So topic model is some sort of generative model. And this just as a model that captures this idea about the collection having topics and topic document having different proportions. So what it says is that. So you are having some documents they are nothing but mixtures of topic and a topic it is nothing, but the probability distribution over words. So what are 3 themes we are talking about?

(Refer Slide Time: 11:55)



We are talking about a collection, that is having documents  $d_1$   $d_2$  up to  $d_n$ . Here a documents in the collection. Then you are having some topics. So suppose there are some  $t_1$  to  $t_k$  topics. And use your documents in document some words will occur and you say these are my vocabulary. These are my words  $w_1$  to some  $w_m$ .

So here 3 themes, in the collection there are documents, documents some words occur. Suppose you have unique words define your vocabulary and there are some topics. Now what topic models say? Yes topics are nothing but probability distribution over words. So  $t_1$  would be something like a distribution. So here word 1 comes the probability of 0.01 word 2 comes with the probability of 0.1 and so on. Similarly,  $t_k$  will be again a probability distribution word 1 comes with a probability distribution of 0.05, word 2 with a 0.01 and so on. An idea is that probably there will be some part of the themes.

So this is my topics. Topics are defined by probability distribution over words. Now what are my documents? Documents are again some sort of probability distribution over topics. So I say  $d_1$ , what is  $d_1$ ,  $d_1$  is having a distribution over these  $k$  topics. So I will say topic one occurs with the probability of 0.05, topic 2 with probability 0.2 topic 3 with 0.5 and so on. So I will say something like topic 3 and topic 2 are more turbulent in this document. And not the other topics and same way I can situate all the documents in some sort of mixture of topics. And this is very important to understand the topic model. So what are my topics - probability distribution over the words; and what are my documents?

Again mixture components or topics or you can offset probability distribution over the topics.

(Refer Slide Time: 14:21)

*Topic Model: Basic Idea*

A generative statistical model that captures this intuition.

**Generative Model**

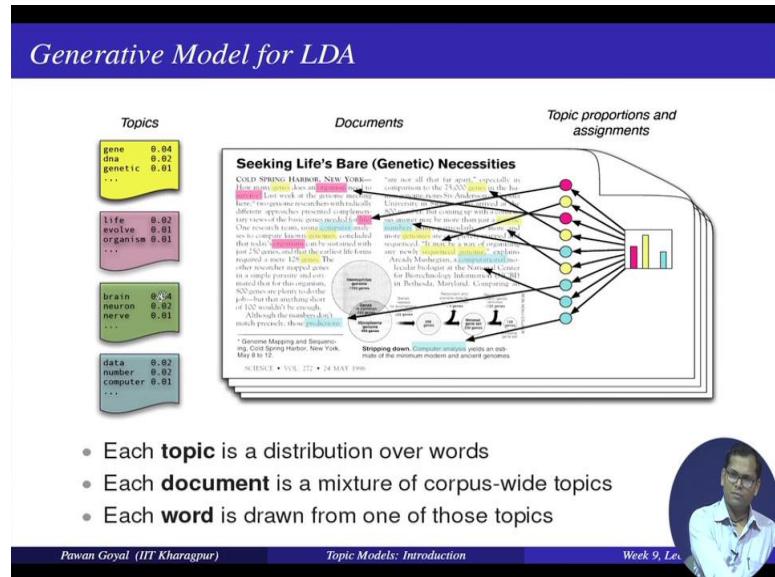
Documents are mixture of topics, where a topic is a probability distribution over words.  
genetics topic has words about genetics with high probability and the evolutionary biology topic has words about evolutionary biology with high probability.

Pawan Goyal (IIT Kharagpur) Topic Models: Introduction Week 9, Lecture 1 7 / 11

So this is all add up to 1, this will all add up to 1 for all the topics, for all the documents. So what is that mean? So you have a topic like genetics, remember no labeling, but we can see when you see that words. So this is probability word genetics. So if we have topic about genetics this will have words about genetics with high probability, and if we have a topical of evolutionary biology, it will have a topic about evolutionary biological of high probability. So this will be making 2 different themes in this connection.

And the model is a generative model and as we understand generative model. So it will work like from we will first generate the topics, yes we first define the topics. Then when we have the topics, you will now start building the documents, you say document will have some proportion of these topics and then I will write the words in the in the documents. So topic search in that the first and then the documents as per the generative model.

(Refer Slide Time: 15:13)



So this picture will make themes clearer. So we are having seen some nice colors here. So, on the left you are seeing some topics right. That is all you are talking about. We are having a set of topics any topic is a probability distribution over the words in the collection. So like here this topic has the word gene with the probability of 0.04, dna with 0.02, genetic with 0.01, life with 0.02. This is different topic brain neuron nerve right data computer number different probabilities. You see seen their different themes.

Now, once we have this corpus wide themes. So each topic is a distribution over words. Do you understand that now what is a document? So you are seeing a collection here and there are a lot of documents, right. And we are being shown one document. So each document is a mixture of corpus wide topics. So these are my corpus wide topics now I will take a particular mixture of these topics. So suppose one of my mixture is here. I take this topic red yellow and blue I will take these 3 topics only maybe others away with a very small fraction. And this defines the topic distribution of my document.

So I have now seen said that my document contains these topics with some proportions. Now how do I generate the words in the documents right. That is important I need to generate the words. So how are these words generated that is again interesting? So you have this you think of it as a multinomial distribution. And from this distribution you sample a topic. Suppose the first sample is this pink that pink topic. That is about an organism life evolve and so on these sample is topic.

Now from the same topic you have to generate a word. How do you generate a word again this topic is what a probability distribution over words? Think of it as a multinomial distribution against a sample a word from here. And that is what you will generate in the document. And this we will keep on going for generating all the words in the document in the in the document. So you will say next word I will again sample a topic I get this yellow topic about gene dna genetics. I use this topic to generate a word by sampling from a multi multinomial distribution. And that is what are you keep on repeating. This is for this document again for next document I will select a mixture of topics. And then once I have the mixture I will again keep on selecting the words that will go in the document.

(Refer Slide Time: 17:51)

*What does the statistical model reflect?*

- All the document in the collection share the same set of topics, but each document exhibits those topics in different proportions
- Each word in each document is drawn from one of the topics, where the selected topic is chosen from the per-document distribution over topics

In the example article, the distribution over topics would place probability on *genetics, data analytics and evolutionary biology*, and each word is drawn from one of those three topics.

Pawan Goyal (IIT Kharagpur) Topic Models: Introduction Week 9, Lecture 1 9 / 11

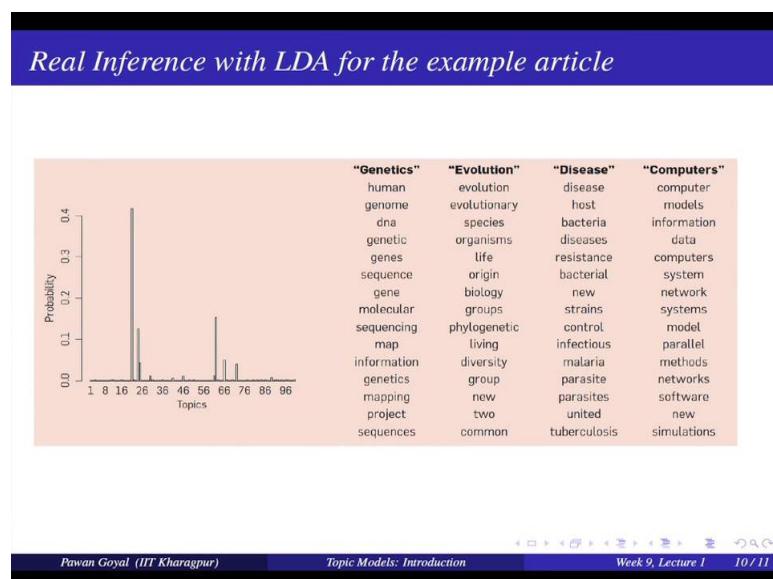
So this is a statistical model also we called it a generative model. So what is this reflect? So it reflects to 2 important things. What are those? All the document in the collection share the same set of topics right. We have an underlying set of topics all documents are sharing the same set, although the proportions are different. So each document exhibits those topics in different proportions. That is one important fact. And then what is the other fact? Each word in each document is drawn from one of the topics where the selected topic is chosen from the per document distribution over topics.

So is that clear? So for each document I have a collection of topics, I have a distribution about topics. So I say this topic is probably 0.2, 0.5, 0.1 and so on. Now each word in the

document is sampled from this distribution. How? You sample a topic first and then as per the topics probability distribution you sample a word. This we will cover in the generative model in more details, but that is the intuition that we showed also from the picture. See there are 2 important facts about LDA.

So if we think about the example article from science magazine that we were seeing. So the distribution over topics would place a probability on genetics data analytics and evolutionary biology. And each word will be drawn from one of these topics. So what will happen? When you are doing the influencing you will say, these this document contains these 3 topics and how do we generate a word you will sample a topic from here and as for the topics probability distribution sample a word and keep on generating the words.

(Refer Slide Time: 19:40)



So this is the genetic model, but is that what you also see when we have a real corpus we apply LDA and do an influencing. So that is what is being shown. So for this collection of science papers. So in proper model was trained with hundred topics and this is what you see. So these are the hundred topics and for this particular document some topics have a high probability. So this topic has probability roughly 0.4 0.15, 0.12 and the next one having 0.5. So if you see the top 4 topics in this particular document and try to see the most probable words in these topics. So that is what you see. So this topic has human genome dna genetics and so on.

Since our topics that we were showing earlier. And you see this was obtained without doing any manual labeling. So you find out this this document contains these 4 different themes important themes. And this is you can also do some labeling later on manually. This is not done by LDA, but get this is also not very important. So what is important is that LDA can help you to obtain this sort of distribution. This in a very uncivilized manner. So you can find out what are the overall themes and for a document what are the most important themes.

(Refer Slide Time: 21:03)

*Central Problem of LDA*

- The documents themselves are observed, while the topic structure - the topics, per-document topic distributions, and the per-document per-word topic assignments - is *hidden structure*.
- The central computational problem is to use the observed documents to infer the hidden topic structure, i.e. *reversing* the generative process.

Pawan Goyal (IIT Kharagpur)      Topic Models: Introduction      Week 9, Lecture 1      11 / 11

So we talked about what is the generative model of LDA. But what is the main problem of LDA? So we are saying we will generate the topic first and then we will generate the documents right by sampling a distribution of topics each for each word I will take a topic it is distribution I will take a word and so on. But that is not how we write the documents right. So how will that we used. So it has to be used in a manner that I am observing some data I have the generative model and now I am trying to estimate the parameters of this genetic model by using my observations that is what parameters will maximize the likelihood of seemed observation. So this is like reversing this whole process of LDA that we are talking about.

So we know the documents object in a collection and if the documents, but I do not do the topic structure. So I do not know what are my topics. So whatever distribution of words within each topic. I do not know for each document what will be the distribution

of topics. And I also do not know for each word in a document what will be the topic assignment. I do not know any of this a prior. So this is all my hidden structure.

So center problem LDA is to reverse in to process and use the observed documents to infer the hidden structure. So what is the hidden structure can you infer that by seeing only the object documents. And this may also be called as some sort of reversing the generative process. And that is a center problem of LDA. So this is this initial introduction lecture was about to give to give you the intuition, but now that you have some intuition of what LDA is what topic model is, we will next go to details about what is the mathematical model of LDA. And how do you solve this problem of reversing the generative process. So that will be covered in more details in the next lecture.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 43**  
**Latent Dirichlet Allocation: Formulation**

Welcome back for the second lecture of this week. So, we had started a formulation of topic models in the last lecture so, but we mainly focused on the basic intuitions. So, today we will do the mathematical formulation of latent Dirichlet allocation. So, that is the main sort of topic models that are used; also known as LDA.

(Refer Slide Time: 00:45)

*Central Problem of LDA*

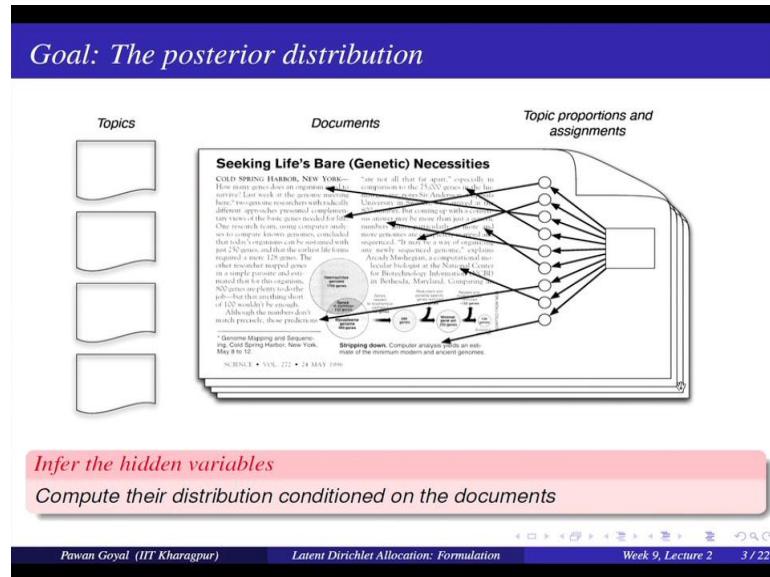
- The documents themselves are observed, while the topic structure - the topics, per-document topic distributions, and the per-document per-word topic assignments - is *hidden structure*.
- The central computational problem is to use the observed documents to infer the hidden topic structure, i.e. *reversing* the generative process.

Pawan Goyal (IIT Kharagpur)   Latent Dirichlet Allocation: Formulation   Week 9, Lecture 2   2 / 22

We were here and we were seeing that what is the central problem of LDA. So, problem we were saying was that we have observing only the documents, but we know nothing about all the parameters of my generative model. So, I do not know, what are my topics? What are the topic proportions of the document and I do not know what is the per document or per word topic assignment, I do not know that.

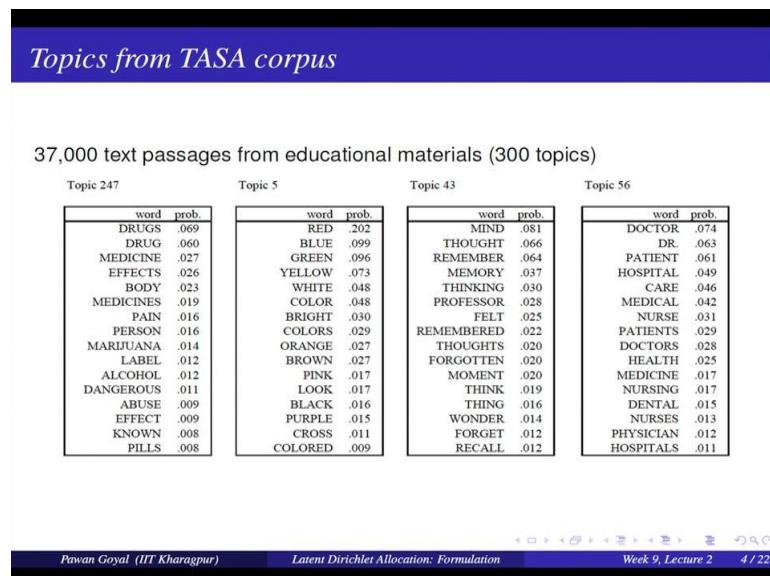
My real problem is from my observation I want to reverse the generative model and come up with all these probabilities.

(Refer Slide Time: 01:24)



So this figure will help you in understanding this goal. So, remember we saw an earlier figure where we had all these topics given to us, we knew what are the proportion; topic proportion for this document and then we were drawing topic for each word in this document, but now so in reality, I will only have the or the documents. So, I know all these documents. So, I know what are the words, but I do not know anything about what are my topics what are these proportions and what are these topics assigner for individual word.

(Refer Slide Time: 02:09)



I have to compute the distribution conditioned on all the documents that I am seeing in my data. So, another simple example to listen the addition behind the generative model, so, here you have some 37000 text passages from some educational material and suppose you run LDA you found roughly 300 topics. So, here in this slide, what you are seeing? You are seeing 4 different topics. So, you are having the topic like 247 which has words like drugs, drug medicine, effects, body, etcetera; another topic having words like red, blue, green, yellow, white, another word about mind, thought, remember, memory, thinking and another topic about doctor, patient, hospital, care, and so on.

Therefore, the topics that are there in the corpus by running LDA over these 37000 text passages, now to get intuition about the generative model. So, what was in the generative model? You have some topics, now how to generate topics? We take some of these topics in some proportions and you start generating words for that. Now, suppose you try that.

(Refer Slide Time: 03:18)

*Topics from TASA corpus*

Documents with different content can be generated by choosing different distributions over topics.

- Equal probability to first two topics: about a person who has taken too many drugs and how that affected color perceptions.
- Equal probability to the last two topics: about a person who experienced a loss of memory, which required a visit to the doctor.

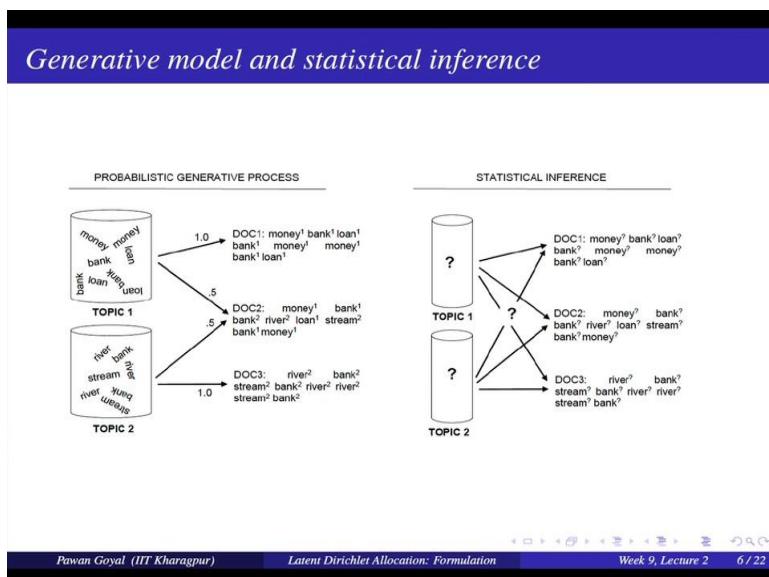
Pawan Goyal (IIT Kharagpur) Latent Dirichlet Allocation: Formulation Week 9, Lecture 2 5 / 22

Suppose you take topic 247 and 5, so, suppose I give an equal probability of the first 2 topics, so, what can a document you can generate? This thing, we will see these topics. So, first topic is about drugs, medicine, person, marijuana, second and another topic is red, blue, green, colors. So, suppose I bind these 2 topics together. So, what kind of document can I generate? So, there can be some different sort of documents you can think of, so one could be that someone who was taking a lot of alcohol marijuana and so on and it affected its color perception.

Suppose you want to document general document like that we will blend these 2 topics together and write that. Similarly suppose you want to blend these 2 topics together. So, this topic is about mind, thought, remember, memory, forgotten and this is about doctor, medical, nurse, patients and so on. So, here if you blend these 2 topics together, you may generate the document that says someone who had suffered some sort of memory loss and it is led to visit of the doctor. So, like that you are having different topics you can blend some of these topics and then generate doc document. So, this is the generative view.

We can give equal probability to the first 2 topics that gives some sort of documents and equal probability to the last 2 topics that give me another sort of document.

(Refer Slide Time: 04:46)



Now this single picture explains both the parts; the generative part and the inference part together. So, what you are seeing in the left figure? So, this is the generative model. So, you are generating some documents and how are you doing that? So, we are having some topics, suppose you are having 2 topics, topic 1, topic 2, each topic is nothing but a distribution over words. So, you are having some words like bank, loan, bank, money, loan. So, these mean this words coming multiple times here. So, these words are having high probability on the other hand second topic is river, stream, bank, river so on. So, these are 2 topics.

Now, using these 2 topics, I am trying to generate my documents. So, how can I generate the documents? I will mix these topics. So, suppose I only take topic 1 and I can have a generate document that contains words like money bank loan and so on, I can just take topic 2 and

generate words like river, bank, stream and so on, but I may also take both these topics instead, say equal proportion and generate document like money, bank, river, loan, stream, bank money. So, what you seen here, in this document to the words are labeled with topic 1 as well as 2 because the words could have come from another topic.

On the other hand in document 1, all the words are labeled with topic 1, doc 3 also all the words are labeled with topic 2 because we have the only topics possible in these documents. So, this is my generative model idea. So, here everything here, you have the topics, you have per document proper proportions and you also have per document per word topic assignment. So, you know the topics for each individual word also.

Now, what is the statistical inference part? So, inference part, you only know your 3 documents. So, you know my doc one contains these words, doc 2 contains these words, doc 3 contains these words especially, but you do not know, what are your topics? And you do not know, what are the proportions of various topics that are represented in each document? So, all these numbers you do not know and you also do not know for each word, what is the topic assignment? And all these you have to infer. So, I hope with this figure, this is clear, what do I mean by my generative model and what is my problem that is to infer all these probabilities of my generative models only from my observation.

(Refer Slide Time: 07:23)

*Important points*

- *bag-of-words assumption*: The generative process does not make any assumptions about the order of words in the documents.
- *capturing polysemy*: The way that the model is defined, there is no notion of mutual exclusivity that restricts words to be part of one topic only. Ex: both 'money' and 'river' topics can give high probability to the word 'bank'.

Pawan Goyal (IIT Kharagpur) Latent Dirichlet Allocation: Formulation Week 9, Lecture 2 7 / 22

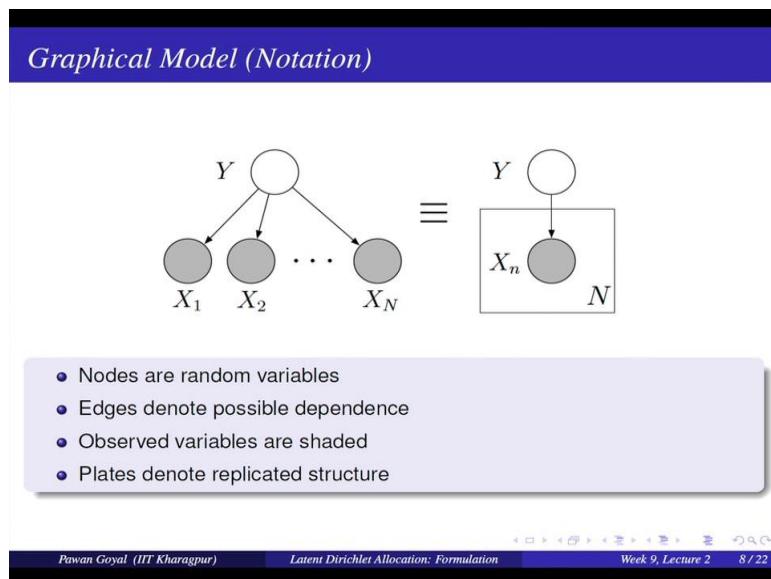
Now important points about LDA, so, first of all it uses a bag of words assumption. So, I hope by now you understand, what is a bag of words assumption? That is I am not looking at

the order in which the words occur. So, this is like a bag forming. So, I am taking all the words and putting them in a set it is not like in list where some order is important. So, LDA does not model the word order as it is. So, it takes only what are the words present in any order.

Second and I guess you would have also noticed that in the previous example, LDAs are also good at capturing polysemy. So, remember what is polysemy? Polysemy is a given word might have multiple senses. So, in the case of topic models, what can I translate that; that means, the same word might correspond to multiple topics and this is perfectly allowed because each topic can have its own probability distribution; that means, the same word can come in 2 different topics also and remember in the previous slide, we are having the word like bank that was coming in both the topics and that is perfectly allowed. So, that way, topic models can capture better this word is coming from topic 1 and topic 2 in its different senses.

The way the model is defined there is no notion of the words being mutually exclusive to the topics. So, for example, money and river can give high probability to the word. So, both money and river topics can have high probability for the word bank and that is perfectly fine.

(Refer Slide Time: 09:03)



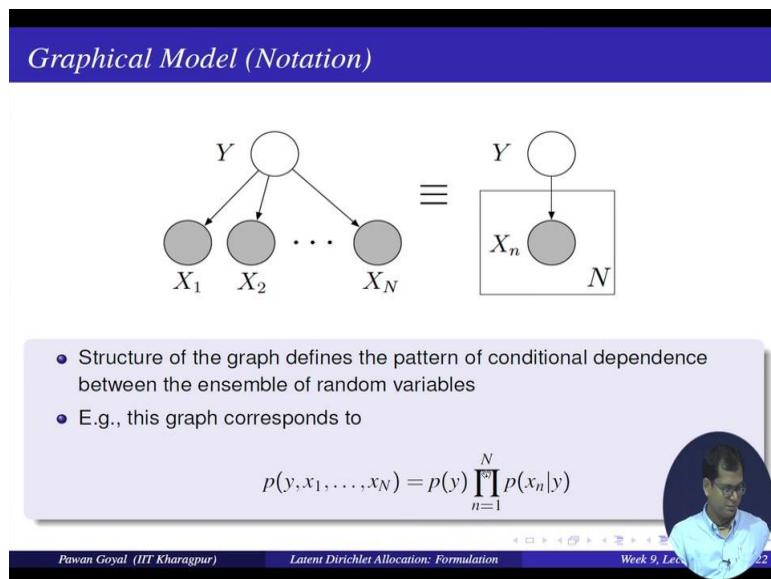
Now, to understand the LDA model, so what is the LDA model? So, you have to first see, what is a graphical notation? So, if you are not gone through some of these graphical notations, so this slide tries to explain, how do we interpret the graphical notations? So, here, what you are seeing? I am having some variables. So, having a variable  $y$  and some variables

$x_1$  to  $x_n$ , so all these nodes when that I see my graphical model, these are random variables. So, I have random variable  $x_1$  to  $x_n$ .

Now, what are these edges? These edges will denote the possible dependence. So, here I know that  $x_1$  depends on  $y$ ,  $x_2$  depends on  $y$  and  $x_n$  also depends on  $y$  and all these depend only on  $y$ , but there is no dependence between each other. So, that is why there is no edge from  $x_2$  to  $x_4$  and  $x_1$  to  $x_2$ . So, these depend only on  $y$ . Then there is a difference that some are shaded; some are not. So, what is that? So, observed variables are shaded so; that means, these are the variables that I am observing and this is a hidden variable that is not shaded. And to simplify these notations, what we can use? We can also group some variables together. So, some max instruction that is being repeated, you can put it in the plate notation. So, this is you are seeing in the right hand side. So, you are having these and different variables and you can group them together by  $x_n$  and you write here capital  $N$ ; that means,  $x_n$  is repeated capital  $N$  times.

This and these are equivalent and these further simplify these notations. So, once we have seen this. So, how do we interpret graphical notations then we can look at the graphical notation for LDA.

(Refer Slide Time: 10:51)



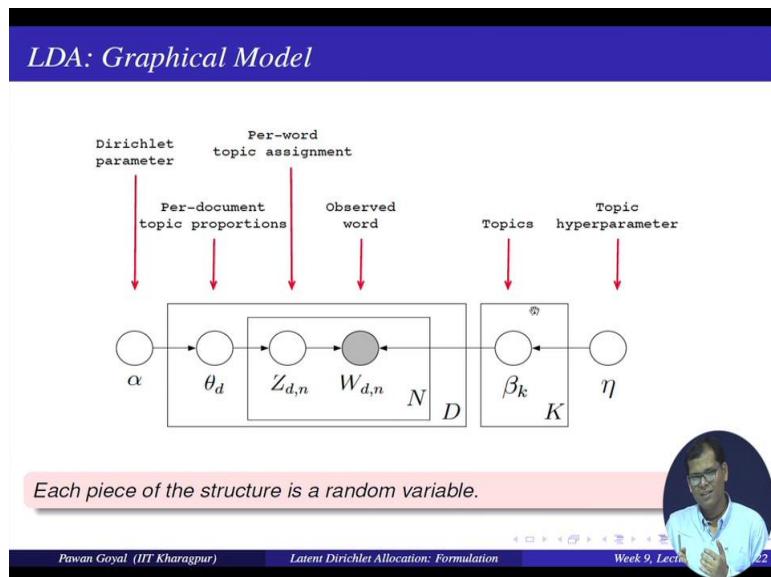
So just one more thing, that once we have this notation, we can also compute the probability of this the whole graphical structure. So, I have variables  $y$   $x_1$  to  $x_n$ . So, probability would be I have the probability of  $y$  and each of these depend only on  $y$ . So, the probability of this

whole structure can be probability of y times probability of x 1 given y x 2 given y up to x n given y.

$$p(y, x_1, \dots, x_N) = p(y) \prod_{n=1}^N p(x_n|y)$$

This graphical; the structure of my graph also defines, what are the conditional dependence between various variables and that I can use to write my joint probability distribution. So, this is my joint probability distribution over all these variables.

(Refer Slide Time: 11:39)



Now let us see, what is the graphical model for LDA? So, you know from the previous slide, how do interpret graphical model? So, all these nodes are random variables and observed variables are shaded. So, the only shaded part is  $W_{d,n}$ , these are my object words, everything else is hidden.

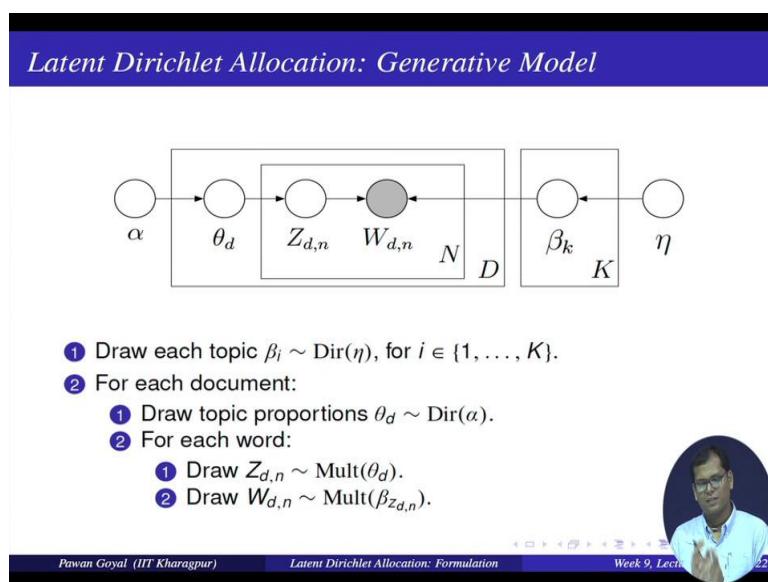
Now, what are everything else? So, there are some plates here. So, let us see this one capital K and what are these topics? So, what I am saying here? There are capital K different topics and each topic is a probability distribution. So, this is the probability distribution over top over different words for a given topic, for a topic it is small k. So,  $\beta_k$  is what is the probability of word one what is the probability of word 2 etcetera for the topic k and this I am doing for all the capital K topics.

Then let us look at this part. So, I have capital D, different documents in my collection. So, for a given document small d, I have figure d variable that denotes per document topic proportions. So, for this document is small d, what are the proportions in which you are blending different topics together? So, this will be different for different topic; different documents. So, you are having capital D, different documents and there is again a Dirichlet parameter here and a topic hyper parameter here. So, we will discuss what are these? But right now, you can understand by simply this beta k is the distribution over words for a given topic and theta d is a distribution over topics for a given document.

Now, let us go inside. So, here, now we are going to 1 particular document and this can have capital N words and what are these? So, you are having per word topic assignment Z n, yes, each word will have only 1 topic and this is the actual word that you are observing and these are all hidden. So, I do not know what my betas? I do not know my theta; I do not my Z n. So, this is only to explain, what are different nodes here? What are the different plates here? So, your plates corresponding to topics documents as well as the words in a single document and you have all the variables that we were using, all the notion that we were using earlier, there are variables for each of these.

Now, let us see how; what is the actual generative model? How we generated the words using this structure?

(Refer Slide Time: 14:26)

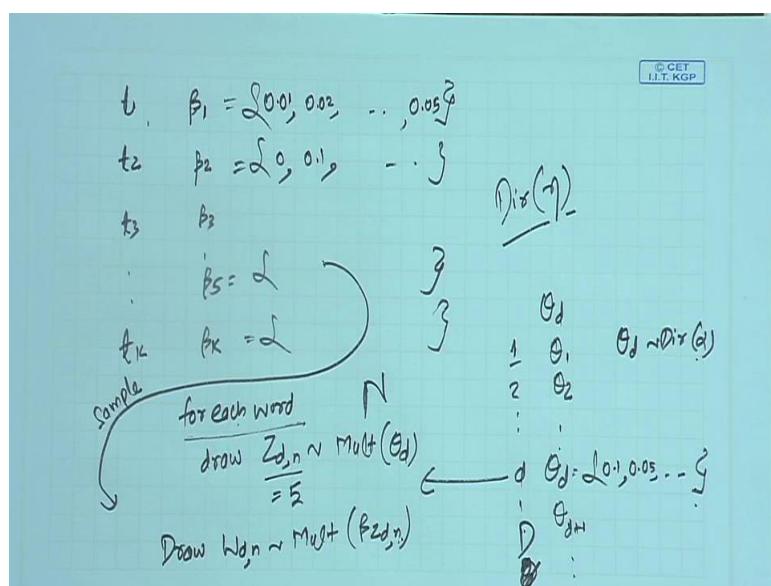


What is the generative model? So firstly, you draw each topic beta  $i$  as for the Dirichlet higher hyper parameter eta and you do that for all the K topics. So, first of all, we generate the model, you are drawing your topics. So, fine I have my capital K topics, now you go to your documents. So, for each document that is small d, draw topic proportions. So, you find out, what are the topics that are involved in this document as Dirichlet distribution over alpha. So, we will talk about these Dirichlet distributions, what do I mean by these Dirichlet distributions?

You draw a topic proportion and then for each word in my observation. So far; I am now going to this document for each word, you draw  $Z_{n,d}$  as a multiple from a multinomial distribution over theta  $d$  and draw a word from this. So, draw a topic and then draw a word from this multinomial distribution over beta  $Z_{n,d}$ .

Now, yeah let us try to understand this a bit more. So, let us go one by one, draw this topic beta  $i$  is your Dirichlet distribution over eta.

(Refer Slide Time: 16:05)



What I am doing here? I have  $k$  different topics. So, I am trying to draw the proportion the probability distributions from my vocabulary. So, what I am saying? So, I have  $k$  different topics; topic  $t_1, t_2, t_3, \dots, t_k$ , for these topics, I am drawing beta 1, beta 2, beta 3 and beta capital  $K$  and what are these? They are nothing but the probability distribution over my words in my vocabulary. So, it can be something like yeah this is 0.01, 0.02, 0.05, so on, this can be 0.1. So, these distributions are drawn by using a Dirichlet over eta.

So, Dirichlet; it is a distribution over distributions. So, it helps you decide, what kind of distribution will be preferred over another, so that we will again discuss. So, this we will discuss later, but the idea is which kind of distribution, you will prefer and this helps you draw some distributions for all the k topics.

Now this you have drawn, now the next line says for each document, draw topic proportion of theta d using Dirichlet over alpha. So, now, these are your topics, but in your collection you have again capital E documents. So, document 1, 2, small d, capital D, so they are capital D documents in your collection, now what is it? What are you drawing for each document? You are drawing theta d, what is theta d? Theta d is a probability distribution over the topics. So, it will be something like what is the probability of topic 1? What is the probability of topic 2? What is the probability up to topic capital D?

For each document, you are drawing these distributions and what are theta d? Theta d are coming from the Dirichlet over alpha. So, again this alpha is telling me, what kind of topic distributions will I prefer over others? So, these are my, so first drawing the topics then drawing the topic proportion for the documents, now what else, now it says now suppose I am going to this document then for each word, how do you about words, draw Z d n as in from a multinomial over theta d. So, theta d is a probability distribution and it is a multinomial distribution from there sample one topic. So, this distribution k topics, suppose the Z n is equal to 5 that is I am taking topic 5. So, for each what I will draw 1. So, it can be whatever that that comes according to this multinomial distribution.

Suppose it is 5, now how do I generate the word? Now I know the topic, now I have to generate a word, so, I go to so it says draw W d n from multinomial of beta Z d n, what is Z d n? Now Z d n knows my 5 that is the topic you that you draw and now you take multinomial over beta 5 so; that means, for beta 5, fifth topic I will find out what is the. So, I will have the distribution and from there I will sample one word. So, I will sample about from this probability distribution and that is what I am generating.

This you will do for each of the capital N words in my document and is the whole generative model first you are generating your topics then for each document topic proportions then you are going to the individual topic going to the individual word and generating that word fine so that is what we were saying in this slide.

(Refer Slide Time: 20:20)

## What is Latent Dirichlet Allocation (LDA)?

- 'Latent' has the same sense in LDA as in Latent semantic indexing, i.e. capturing topics as latent variables
- The distribution that is used to draw the per-document topic distributions is called a *Dirichlet distribution*. This result is used to allocate the words of the documents to different topics.

**Dirichlet Distribution**

The Dirichlet distribution is an exponential family distribution over the simplex, i.e. positive vectors that sum to one

$$p(\theta | \vec{\alpha}) = \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \prod_i \theta_i^{\alpha_i - 1}$$

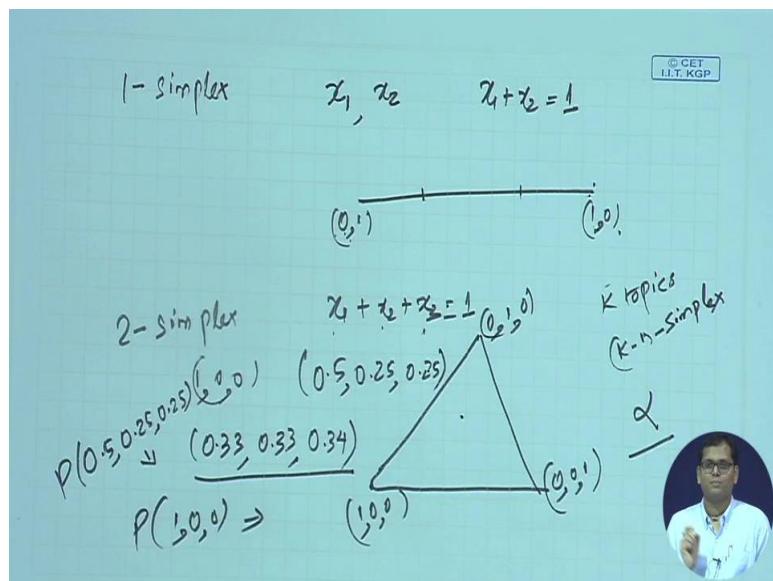

Pawan Goyal (IIT Kharagpur)      Latent Dirichlet Allocation: Formulation      Week 9, Lecture 22

Now so 1 interesting point here is why do we name it at latent Dirichlet allocation? So, we are doing some allocation what why latent and Dirichlet coming. So, latent in this LDA, say has a same sensing energy latent semantic indexing. So, that is all these topics are some sort of latent variables. So, there are some sort of hidden topics, some latent topics, I do not know, what are these topics as such? I cannot give them good maybe good labels also, but there are some distributions some hidden distributions over my words and these are called latent.

Then what is Dirichlet here? So, you saw the Dirichlet distribution at 2 places. So, that is the distribution that is used to draw the per document topic distributions is a Dirichlet distribution. So, that is how am I sampling topics for a given doc document that is coming from a Dirichlet distribution and this result is used to allocate the words of the documents to different topics and that is why the word allocation is also coming. So, you are having latent Dirichlet and allocation.

Now we will look at this Dirichlet part in bit more in bit more details. So, what are the Dirichlet distributions? So, if you think about it, so if you incredibly try to understand that this is a distribution over probability distributions what do I mean by that? So, Dirichlet distribution is an exponential family distribution over the simplex that is the positive vectors that sum to 1.

(Refer Slide Time: 22:11)



When I talk about simplex so you can talk about say 1 simplex would be 2 elements say  $x_1 \times x_2$  such that  $x_1$  plus  $x_2$  is equal to 1 positive vector that added to 1. This is one simplex. So, you can see that they are infinite solutions here and you can read it as a line, it is a line and this might correspond to say 0 1, this might be 1 0. So, that is topic 1 has 0 topic to each or yeah  $x_1$  is 0;  $x_2$  is 1, here  $x_1$  is 1,  $x_2$  is 0 and any point you can accordingly give some definition what are the values of  $x_1 \times x_2$ .

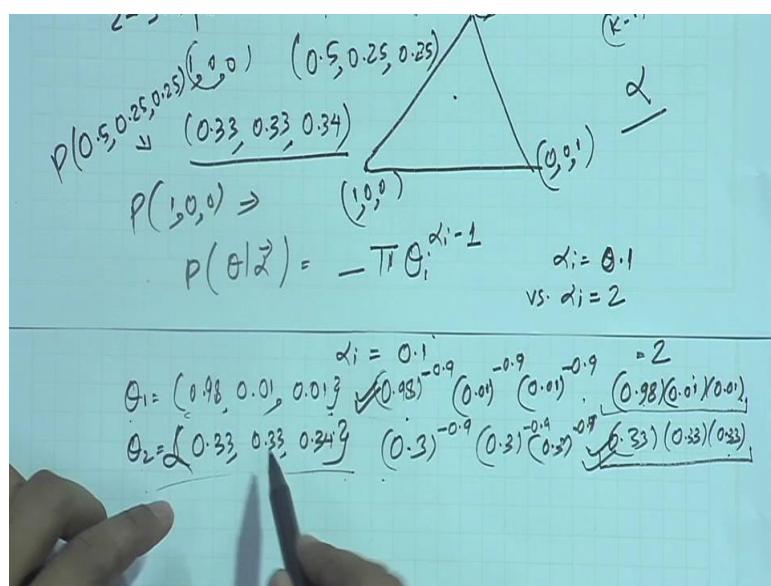
You can see that this will be a line, simple line  $x_1$  plus  $x_2$  is equal to 1, this is my 1 simplex then you can have 2 simplex then you are having 3 things  $x_1$  plus  $x_2$  plus  $x_3$ , there are 2 1, in this would be some sort of triangle. So, this point might correspond to say  $x_1$  is 1,  $x_2$  is 0,  $x_3$  is 0. So, this might be  $x_1$  is 0,  $x_2$  is 1,  $x_3$  is 0. This might be 0 0 1 and then each point, we will have some proportion such that all 3 add up to 1, this is my 2 simplex like that you can define of any simplex. So, if you are having  $k$  topics you can think of it as  $k$  minus 1 simplex.

Now, what is my Dirichlet distribution? Now, we are saying it is an exponential family distribution over the simplex that is a positive vectors that up add up to 1. So, again let us try to understand that intuitively first. So, what I am saying here? Suppose I am having a 2 simplex then lot of different value is my  $3 \times 1 \times 2 \times 3$ , they can take a lot of different values. So, I can take values like 1 0 0, I can take values like 0.33, 0.33, 0.34 and things like that they can take a different values like 0.5, 0.25, 0.25, etcetera.

So, what my Dirichlet distribution does? It gives me a probability of this probability distribution. So, what is the probability of getting distribution like this? What is the probability of getting a distribution like this? So, that is what kind of distributions I will prefer. So, I can use that to say that I will prefer distributions where one topic one of the topics as a high probability and others have very low probability or I will like to have a distribution where all the topics have equal probability.

This kind of constraints you can put by using your alpha. So, that is where the formulation is. So, what is the probability of theta that is a distribution over these topics given my alpha that is some drawn function? So, even if we forget about this term. So, this is multiplication over theta I to the power alpha I minus 1. So, let us look at this term only. So, what is being said here?

(Refer Slide Time: 26:02)



Probability theta given alpha is multiplication over theta i to the power alpha i minus 1. So, let us try to understand that suppose my alpha i is say 0.1 versus alpha i is equal to say 2, what would happen in the 2 cases and let us say I look at 2 different thetas. So, my theta 1 is 0.98, 0.01, 0.01 topic has a high probability others have roughly 0 and theta 2 is say 0.33, 0.33, 0.34 and now you can see what kind of alpha will prefer 1 theta over another 1.

Let us take the case with 0.1 alpha i 0.1. So, what would you is the probability of this getting this distribution it will be 0.98 to the power minus 0.9, 0.01 to the power minus 0.9 0.01 to the power minus 0.9 and this probability would be same 0.3 to the power minus 0.9 0.33 to

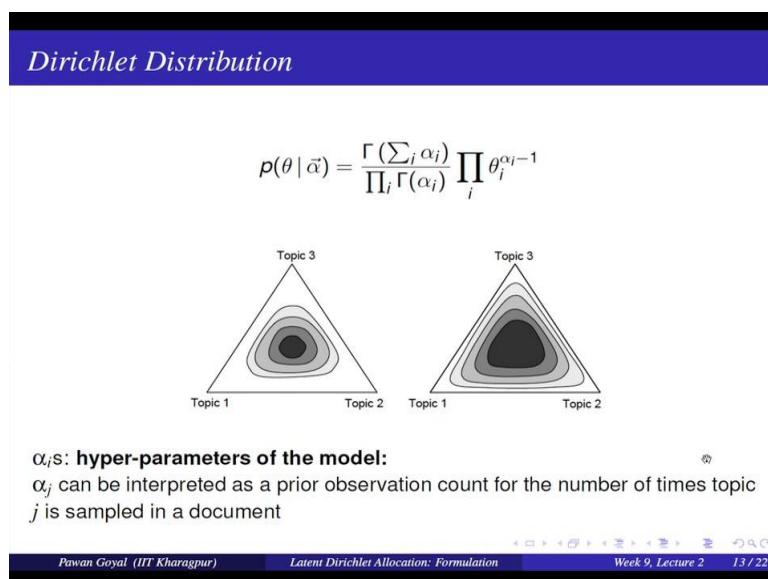
the power minus 0.9 and so on. On the other hand, if I take alpha i is equal to 2 then this would be 0.98 times. So, alpha i minus 1 will become 1 the power is 1.01 times 0.01 and this will become .033 times 0.33 times 0.33.

Now, what is your observation here? So, one thing we see it that if you take alpha is equal to 2, if you take alpha is equal to 2 then the topics where so the distribution where one topic is having very small probability we will get a overall very small probability, your multiple 0.98 by 0.01 times 0.01, this will become very small only when this will be this is like 1 by 3 times 1 by 3 times 1 by 3. So, as you increase alpha, it will prefer to have topics or the distributions where each topic the word probability is roughly equal.

It will prefer this 1, but if you having a smaller alpha then what you are seeing? So, now, 0.01 to the power 0.9, this will be now can written as 100 to the power 0.9, this will become very large and this will be roughly this will not be large. So, what will happen as your alpha becomes small they will prefer the probability distribution where one topic is having high probability and others are having low probability?

By tuning your alpha, you can prefer 1 topic probability, one sort of distribution over the distribution.

(Refer Slide Time: 29:59)



This is my Dirichlet distribution. Now, again to give you some visualization, so here are 2 different sort of simplex. So, alphas as such, you can interpret it as some prior observation

count on the number of times a topic j is sampled individual alpha j so that is how many times this topic will be sampled?

(Refer Slide Time: 30:23)

### Dirichlet Distribution

$$p(\theta | \vec{\alpha}) = \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \prod_i \theta_i^{\alpha_i - 1}$$

$\alpha_i$ s: hyper-parameters of the model:  
These priors can be interpreted as forces in the topic distributions with higher  $\alpha$  moving the topics away from the corners of the simplex

Pawan Goyal (IIT Kharagpur) Latent Dirichlet Allocation: Formulation Week 9, Lecture 2 13 / 22

But and you can also think as some forces and higher alpha will move the topics away from the corner of the simplex. So, let me come back to this point again. So, you are saying 2 different simplex, one where the topics are moved away from the corners, these are being moved away and here you are going towards the corners and this is for because of higher values of alpha and we will come back to this again.

(Refer Slide Time: 30:51)

### Dirichlet Distribution

$$p(\theta | \vec{\alpha}) = \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \prod_i \theta_i^{\alpha_i - 1}$$

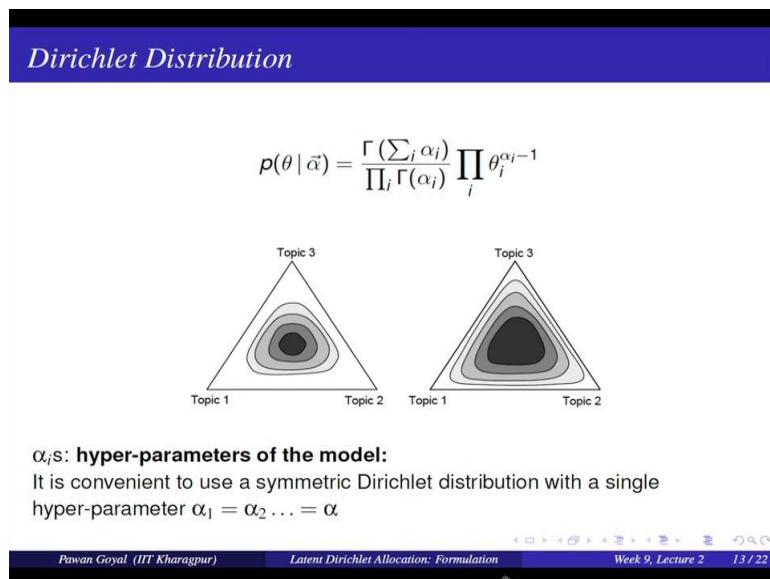
$\alpha_i$ s: hyper-parameters of the model:  
When  $\alpha < 1$ , there is a bias to pick topic distributions favoring just a few topics

Pawan Goyal (IIT Kharagpur) Latent Dirichlet Allocation: Formulation Week 9, Lecture 2 13 / 22

Now when alpha is less than 1, there is a bias to big topic distribution is favoring just a few topics and this is what we saw just now on paper that when alpha is equal to is less than 1, it tends to prefer the distributions where only a few topics as a high probability and others have lower probability.

Now, while in general, you can take different alphas for your different topics, what is convenient effect? You take all help us to be rough to be the same.

(Refer Slide Time: 31:30)



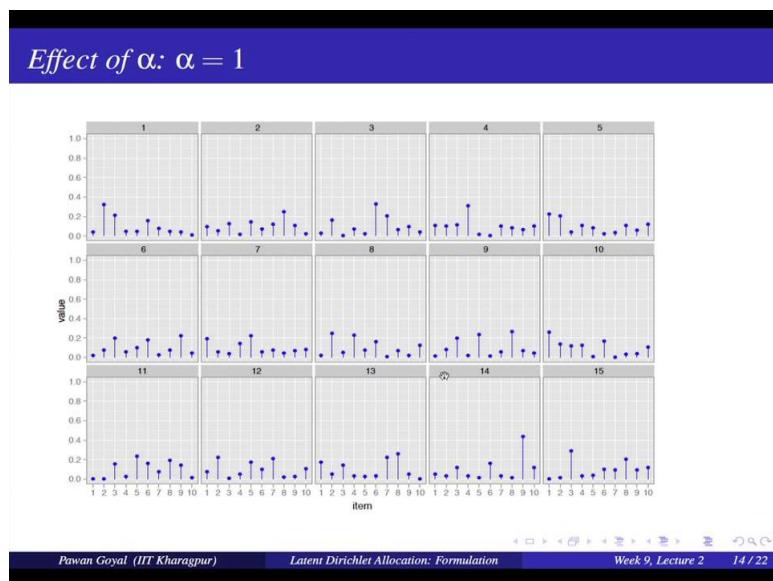
You have a singular hyper parameter alpha. So, I will all alphas are same now, what matters now is what is the value of this alpha? So, the relatively they are the same all alpha 1 to alpha and alpha capital D for all the documents, they are the same sorry alpha 1 to alpha call capital K, they are same, but what is the relative number is it like what is the number is it like 0.125, there will be a matter.

Now if alpha is small, what will happen? You will tend to prefer topics with way which sorry, you will tend to prefer distribution where 1 topic will have a higher probability. So, remember how do we define simplex? So, in this case what this it is boundary means. So, these colors denote, what is the probability distribution? So, black means a high dist probability distribution and so on and as it diminishes so the color becomes so faded then, you are seeing that the probability is decreasing.

In the left hand side figure, the probability is mostly centered for in the center of the simplex by in right hand side, it is moving towards the corners. So, that you can interpret it as if in this simplex whenever all 3 topics have the same weight it is given a higher probability here even if one topic is having in more proportions than others it is getting a high probability. So, this is not moving away from the center of this simplex.

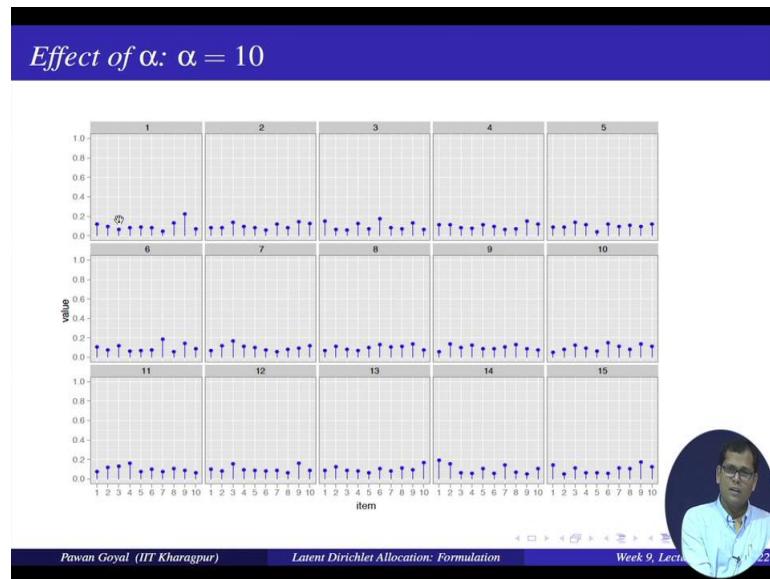
It is going towards the corner. So, if you go to the corner; that means 1 topic has the high probability, other 2 have the small probabilities. So, this is favoring also to have topics where sorry, also to have distributions where one topic has high probability than others while this we favor the distributions where all 3 topics have roughly equal probability. So, now, you can easily tell which one is corresponding to higher alpha lower alpha. So, the one that corresponds to lower alpha will be like that it will favor distributions where 1 topic is having a high probability than others.

(Refer Slide Time: 33:57)



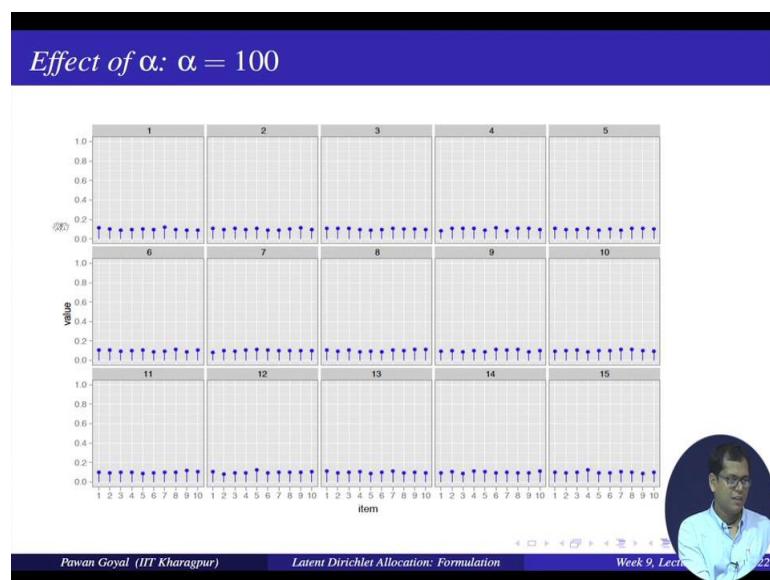
Now, this is some from simulation, what happens if you take different values of alpha? So, if you take alpha is equal to 1, what kinds of distributions are preferred. So, there are 15 documents here that are being shown and there are 10 different topics.

(Refer Slide Time: 34:27)



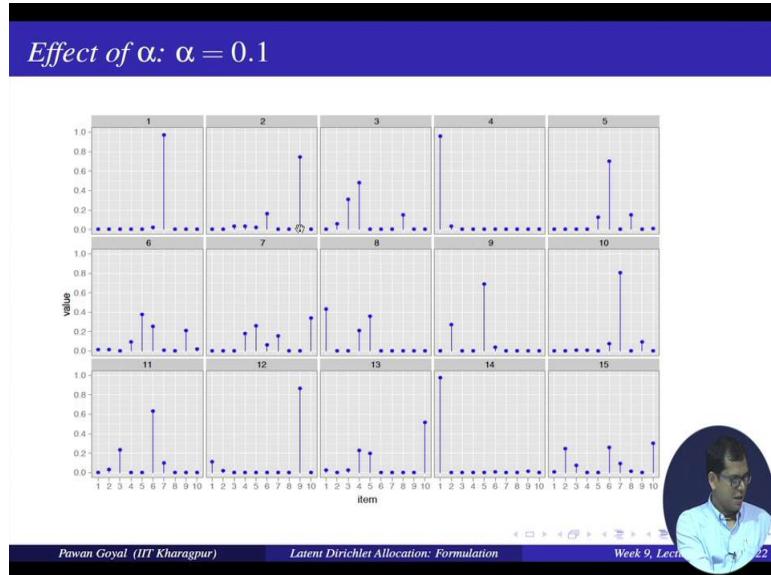
We see the distributions are so you are having some distribution for topic t 2 t 3 and so on for these 15 documents, they are different different distributions, but suppose you try to now increase the value of alpha as you go to 10. So, as you increase the value of alpha, it will start favoring those distributions where all the topics are roughly same, probability you see now this is getting flattens. So, you are having, you are now seeing all the topics and if you try to increase alpha to 100, you will see all the topics have same probability and it is not something that you will there.

(Refer Slide Time: 34:49)

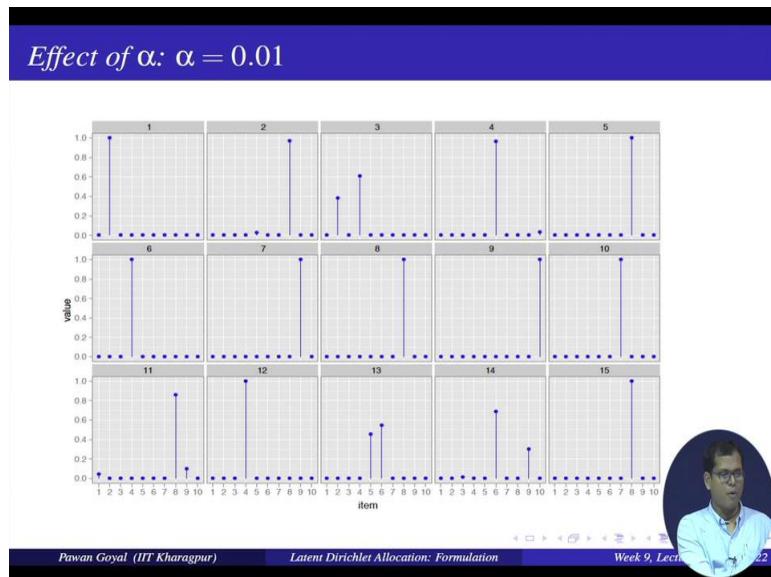


It is not good that each document has all the topics in same probability then if the topic model is does not put in any sense. So, alpha is equal to 1 was looking ok.

(Refer Slide Time: 35:12)

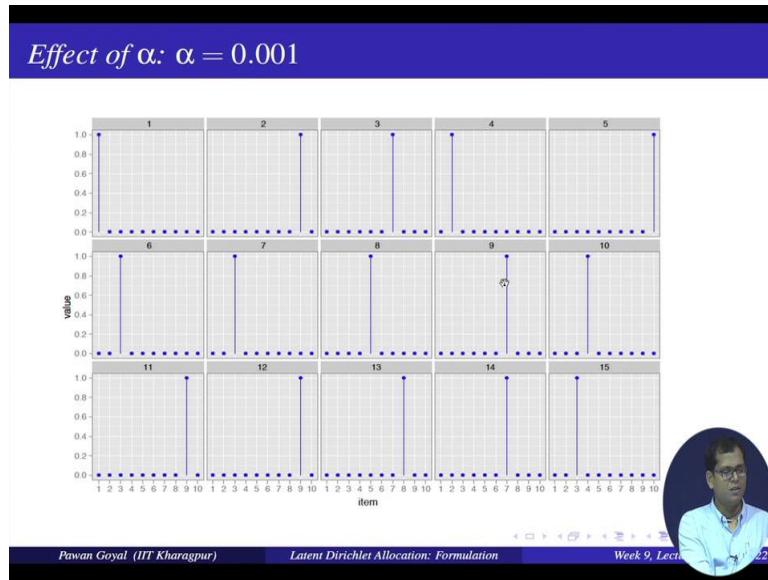


(Refer Slide Time: 35:24)



This kind of distribution we would like, but now suppose I want to decrease the value of alpha, if suppose I go from 1 to 0.1, now what you are seeing? It will start favoring the distributions where one topic has a high probability or 1 or maybe 2, but suppose I increase, I decrease it further to 0.01.

(Refer Slide Time: 35:34)



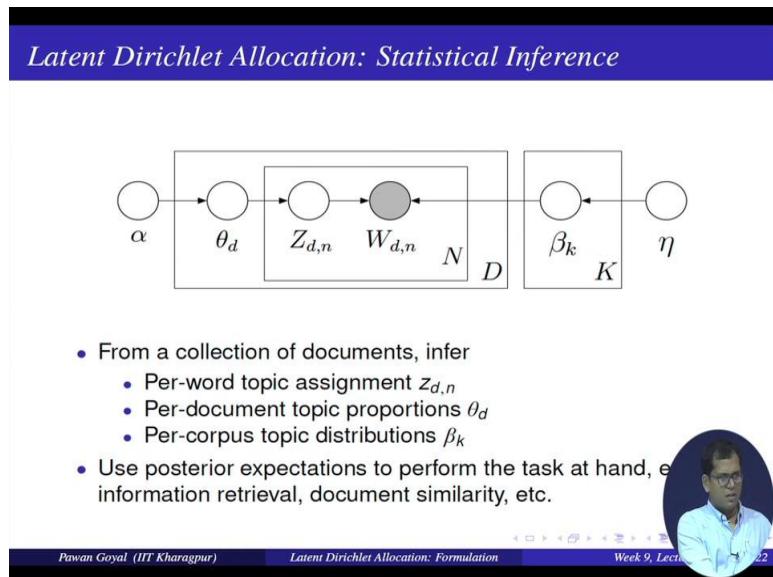
Most of the problems of probability 0, only 1 topic comes as a probability 1 or here 2 topics are coming. If you further reduce this value to 0.01, you get only 1 topic in each document. So, that will give you some idea of how if you modify or change this parameter alpha, how does this affect the overall topic distribution in my corpus, certain kind of distribution, we will get reference over others.

(Refer Slide Time: 35:54)

Online Implementations	
<b>LDA-C*</b>	A C implementation of LDA
<b>HDP*</b>	A C implementation of the HDP ("infinite LDA")
<b>Online LDA*</b>	A python package for LDA on massive data
<b>LDA in R*</b>	Package in R for many topic models
<b>LingPipe</b>	Java toolkit for NLP and computational linguistics
<b>Mallet</b>	Java toolkit for statistical NLP
<b>TMVE*</b>	A python package to build browsers from topic models

Now, for LDA, there are a lot of interventions that are available so and these are like very very popular, you can also use implementations that are available in gensim.

(Refer Slide Time: 36:06)



Now so we are again, so we have discussed all the in genetic part, but now I am full details that how do we; what is the generative model of LDA? But remember, what is your problem? How do we infer all these probabilities? So, that is I am given a collection of documents and I want to infer per document topic assignment  $Z_{d,n}$  per document topic distributions  $\theta_d$  and per corpus topic distributions  $\beta_k$ , I want to infer all this.

Now, once I am able to infer all this, I can use this to find out say to use to for information retrieval, document similarity and many other task, but the question is once I am given these observations of the words, how do I infer all these probability values. So, there are different ways of doing that. So, we will discuss about one such method in the next lecture.

Thank you.

**National Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 44**  
**Gibbs Sampling for LDA, Application**

Hello everyone. Welcome to the third lecture of this week. So, we were talking about topic models and we have discussed; what is the formulation of latent Dirichlet allocation; and we say that what are the different parameters that we need to learn. We need to learn mainly three parameters that is what are my topic distributions given a topic what is the probability of each word, what are my per document topic proportion, so that is my thetas and then per document per word topic assignment that is by  $z$ . So, this I have to compute the posterior distribution of all this parameter given my observation, observation is all my corpus all the documents that I am seeing.

So, in this lecture, we will discuss one interesting method for doing that is Gibbs sampling. And then in the end, we will also talk about some simple applications that once we had learnt topic models over a corpus what are the some of the simple things that you can do.

(Refer Slide Time: 01:21)

The slide has a dark blue header bar with the title "Approximating the posterior". The main content area is white with a dark blue footer bar containing navigation icons and slide details.

**Approximating the posterior**

Algorithms to approximate it fall in two categories:

**Sampling-based Algorithms**

Collect samples from the posterior to approximate it with an empirical distribution

**Variational Methods**

- Deterministic alternative to sampling-based algorithms
- The inference problem is transformed to an optimization problem

Pawan Goyal (IIT Kharagpur)    Gibbs Sampling for LDA, Applications    Week 9, Lecture 3    2 / 13

So, for approximating the posterior probabilities of all these parameters, the algorithm generally fall in two categories. So, one are sampling based algorithms so that is from the posterior you try to collect the samples of the distributions, and then approximate it with the

empirical distribution and that is what we will focus on in Gibbs sampling. And there are also variational methods, so where we convert the inference problem to some sort of optimization problem and try to learn the parameters that we optimize that. So, both the methods are very much used in the literature. So, we will see Gibbs sampling that is an easier to understand method that I will say.

(Refer Slide Time: 02:30)

The slide has a blue header bar with the text "Gibbs Sampling". The main content area contains a list of bullet points:

- A form of Markov chain Monte Carlo (MCMC), which simulates a high-dimensional distribution by sampling on lower-dimensional subset of variables where each subset is conditioned on the value of all others
- Sampling is done sequentially and proceeds until the sampled values approximate the target distribution
- It directly estimates the posterior distribution over  $z$ , and uses this to provide estimates for  $\beta$  and  $\theta$

At the bottom of the slide, there is a video player interface with a circular video thumbnail showing a person speaking, and text labels: "Pawan Goyal (IIT Kharagpur)", "Gibbs Sampling for LDA, Applications", and "Week 9, Lecture 1".

So, Gibbs sampling are some sort of Markov chain Monte Carlo methods. So, what is the idea? So having a high dimensional distribution; think about all the possible values that that your parameter can take all your betas, theta, z, they can take so huge number of values. So, idea here is you sample on lower dimensional subset of variables and each subset is conditioned on whatever as it known.

So, you do not computing the joint probability of everything you are assuming sum to be known and computing probability for others and that you keep on doing in terms for all the all the variables. So, sampling is done sequentially and proceeds until the sampled value is approximate my target distribution. And it directly estimates the posterior distribution over z, and uses this to provide estimates for beta and theta. So, we will find out the distribution over z, and then use that to compute my theta and beta and we will see how do we do that.

(Refer Slide Time: 03:12)

*Gibbs Sampling*

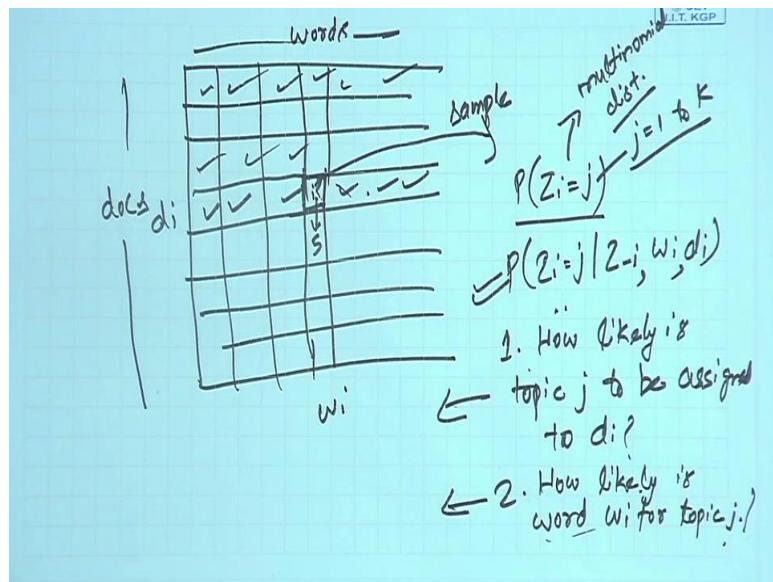
- Suppose we have a word token  $i$  for which we want to find the topic assignment probability :  $p(z_i = j)$
- Represent the collection of documents by a set of word indices  $w_i$  and document indices  $d_i$  for this token  $i$
- Gibbs sampling considers each word token in turn and estimates the probability of assigning the current word token to each topic, conditioned on the topic assignment to all other word tokens
- From this conditional distribution, a topic is sampled and stored as the new topic assignment for this word token
- This conditional is written as  $P(z_i = j | z_{-i}, w_i, d_i, \dots)$

Pawan Goyal (IIT Kharagpur)    Gibbs Sampling for LDA, Applications    Week 9, Lecture 1

So, what is the idea? So, assume that you are having a corpus; and in the corpus you have some documents and words. Now take so you will assume that in a in one of the iteration you are at a particular word that is the word token  $i$ . And what you want to find out what is the topic assignment probability, what is the probability that is  $i$ th token will be assigned to topic  $j$  probability  $z_i$  is equal to  $j$   $P(z_i = j | z_{-i}, w_i, d_i, \dots)$  that is what you want to find out. So, now, how do you do that? Now, you represent the collection of documents by a set of word indices  $w_i$  and document indices  $d_i$  for this token  $i$ . So, it simply says that you have a lot of words find out what is the corresponding word token for this  $i$  that will be called  $w_i$   $w_i$  will give you the particular index and similarly  $d_i$ ,  $d_i$  will be the particular document where you are having this token.

So, what Gibbs sampling does it consider each word token in turn and estimates the probability of assigning the current word token to each topic conditioned on the topic assignment of all other word tokens. So, what it is saying? That in the particular iteration assumes that you the topic assignment for all the other words, except this word. Now, based on this you try assign a topic to this word.

(Refer Slide Time: 04:38)



So, what will happen, you are having a set of documents. So, these are your documents and suppose you know your unique words these are your words. And you are at a  $i$ th token. So, the corresponding document will be called  $d_i$ , this is  $i$  token and the corresponding word will be called  $w_i$ . So, now what you have to find out probability that the topic assignment for this word will be  $z$  that is what we have to find out. And what is assumed is that in this iteration you know the topic assigned for everything else.

So, you know what are the topics that assigned for all different words you know all that, but you want to estimate to find out the topic for this particular  $i$ th word. So, how do you do that, you first estimate this probability and this you do condition on everything else. So, we write it like this probability  $z_i$  at  $j$  given the topic assignment for all the words other words minus sign is everything other than this  $i$  and what is the word index and what is the document index, this is what we want to find out.

Now, intuitively what should it depend on? So, what is the probability that this  $i$ th token will be given assigned the topic  $j$ . What should it depend on? So, we think in terms of topic models, this probability that the  $i$ th word should be assigned the topic  $j$  should depend on two things one is so we say that document consist of certain topics. So, if this topic is prevalent in the document, there might be a high chance that this word is assigned this topic  $j$ , so that is how likely is topic  $j$  to be assigned to  $d_i$ . What is the next thing, how likely is that this word occurs in topic  $j$  is word  $w_i$  for topic  $j$ . So, we have two things I need to know. Now, how do I know

this one, how likely is topic j to be assigned to document d i. You see I am given the topic assignment for all other words in this document. So, I will find out what sections of words in this document are assigned this topic j divide by the number of words. So, this I can computationally by this topic assigned.

Now, how do you compute this how likely each word w i for topic j, I will again see in topic j what are the different words that come and how many times word w i comes in this topic j. And this I can used to compute this probability. And then I can multiply these two to find out what is the probability of a topic j been assigned to this word token i and this I will do for all topics j is equal to 1 to k. So, this will give me a multinomial distribution. Then to give a assignment to this word I will sample from this multinomial distribution.

And then from sampling, I will assign one topic and I will assign the topic here. So, suppose the topic is 5, I will put this is 5. And then I will move to the next word token, and then I will assume that this is not given, and I will take everything as given do that find the topic assignment and that is what I keep on doing. So, now once the intuition is clear, let us go back to the formulation. So, yes from this conditional distribution, you are sampling a topic and we are storing it as the new topic assignment for the word and this is written as this assignment probability that ith word has the has a topic j given everything else.

(Refer Slide Time: 09:08)

### Gibbs Sampling

- Let us define two matrices  $C^{WT}$  and  $C^{DT}$  of dimensions  $W \times T$  and  $D \times T$  respectively.
- $C_{wj}^{WT}$  contains the number of times word  $w$  is assigned to topic  $j$ , not including the current instance
- $C_{dj}^{WT}$  contains the number of times topic  $j$  is assigned to some word token in document  $d$ , not including the current instance

$$P(z_i = j | z_{-i}, w_i, d_i, \cdot) \propto \frac{C_{wj}^{WT} + \eta}{\sum_{w=1}^W C_{wj}^{WT} + W\eta} \frac{C_{dj}^{DT} + \alpha}{\sum_{t=1}^T C_{dj}^{DT} + T\alpha}$$

- The left part is the probability of word  $w$  under topic  $j$  (How likely a word is for a topic) whereas
- the right part is the probability of topic  $j$  under the current topic distribution for document  $d$  (How dominant a topic is in a document)

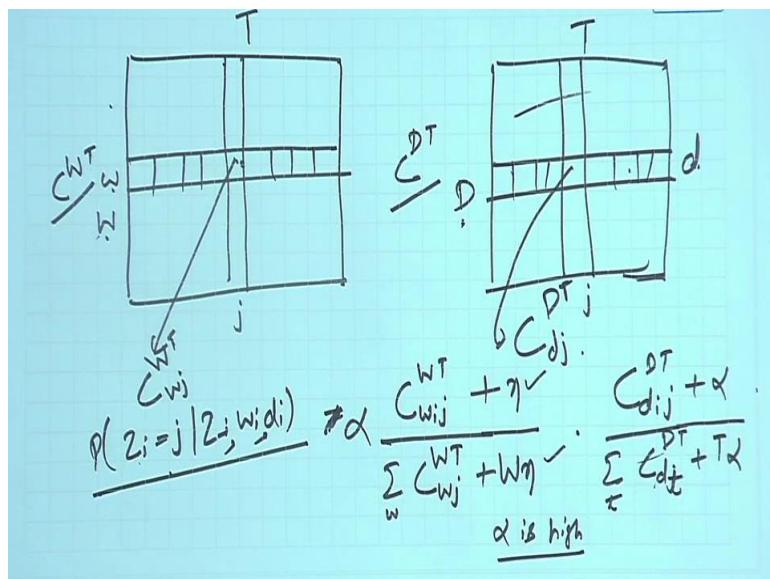
Pawan Goyal (IIT Kharagpur)   Gibbs Sampling for LDA, Applications   Week 9, Lecture 3   5 / 13

$$P(z_i = j | z_{-i}, w_i, d_i, \dots)$$

$$\propto (C_{wd} WT + \eta)(C_{d,j} DT + \alpha) / (\sum_{w=1}^W C_{wj} WT + W\eta)(\sum_{t=1}^T C_{dj} DT + T\alpha)$$

So, how do we achieve this, how do we compute all these values. For that I need to keep two different matrices. So, one is C W T of dimensions W times T another is C DT of dimensions D times T.

(Refer Slide Time: 09:29)



So, what are these matrices? So, one is C W T W times T, and next is C D T D times T. T is the number of topics, W is the number of words and D is the number of documents. So, what do they store? So, C w j, W T that is I take the word w and topic j, so w j W T. So, this element is called C w j W T this contains the number of times word w is assigned to topic j not including the current instants. So, we are taking at the current instants except that how many times this word is assigned this topic that you can find out from the your whole data, you will actually keep this stable. So, we will just update the values.

So, at any point, you know how many times this word is assigned topic j. So, we will do that for all the values in this matrix. How many times this word is assigned this topic? Yes, similarly an element C D T small d j, this will contain for the document d topic j number of times topic j is assigned to some word token in document d that is how many words in the document d are assigned to topic j in that again you can have all the values. So, this will again be not included in the current distance, we will find out how many times any word in document d is assigned

this topic and we will do it for all the documents. So, this you will have these two matrices  $C$   $W T$ ,  $C D T$  and you understand now what is element is.

Now once you have these two matrices how do I compute probability  $Z_i$  is equal to  $j$  given  $z$  minus  $i w i$ ,  $d i$  and research may depend on two different things. So, let us see that. So, depends on two parts, one is what is the probability of word  $w$  under topic  $j$ . Remember we are talking about two parts. So, what will depend on? So, we are saying that it will depend on how likely is this word to come under topic  $j$ ; second is how likely is this topic to be assigned to document  $d$  - two things. How likely is this word to come on to topic  $j$ ? How likely is this topic to come under document  $d$ ?

Now, how do I write this probabilities in terms of this matrix elements how likely is word  $w$  to come under topic  $d j$ . So, that will be  $C$   $i$   $w$   $j$   $W T$  summation over all the words. So, this will give  $p$  probability for this word, summation over all words  $C w j W T$ . Similarly, for how likely is topic  $Z$  to come on the document  $d$  this would be  $C d i$  for the current instance  $j$   $D T$  summation over, now for all the topic that are assigned in this document. So, this will be summation over all my topics  $C d t$ ,  $C d t$  capital  $D T$ . So, this will be the simple formulation.

So, there are certain priors that you take here. So, you will have some sort you can these are some sort of smoothing. So, here you have some smoothing parameter  $\eta$ , similarly it will be plus  $W$  times  $\eta$ , here  $\alpha$  plus  $T$  times  $\alpha$ , so smoothing parameters. And this will again to make it a probability; it is normalized, so instead of calling it equal to you will say proportional to this. So, now, this gives you the formulation probability that the  $i$ th word token will be assigned topic  $j$  given all this.

So, now, you can say that what are these  $\eta$  and  $\alpha$ . So, this is my Dirichlet parameters that we were seeing earlier. And you can also correlate this with their values. So, suppose your  $\alpha$  is high, if your  $\alpha$  is high, what would happen? This value will not matter and all the topics will be assigned roughly equal probability and that is what was happening. If you keep on increasing  $\alpha$  you are going towards a distribution where all topics have some probabilities, but if you  $\alpha$  is very, very small then what will matter is only this count and that is why you are going towards only a few topics. So, these are the intuitions.

Same thing you can do with  $\eta$ . So, if  $\eta$  is high, all the words will have equal probability of coming into topic if it is low then certain distributions will be preferred. So, now this is the formulation for probability that  $Z_i$  is equal to  $j$ . So, left part here is the probability of word  $W$

under the topic j, so that is how likely the word is for a topic; and the right part is the probability of topic j under the current topic distribution for the topic. And this is what we had seen also in the previous. So, when we are doing it on the paper in the previous slide. Now, this will give you the probability distribution over all the topics given this token. Now, what is the next thing, you have this multinomial distribution, you sample a topic from here.

(Refer Slide Time: 16:11)

*Algorithm*

- Start: Each word token is assigned to a random topic in  $[1 \dots T]$
- For each word token, a new topic is sampled as per  $P(z_i = j | z_{-i}, w_i, d_i, \cdot)$ , adjusting the matrices  $C^{WT}$  and  $C^{DT}$
- A single pass through all word tokens in the document is one *Gibbs sample*
- After the burnin period, these samples are saved at regularly spaced intervals, to prevent correlations between samples

Pawan Goyal (IIT Kharagpur)      Gibbs Sampling for LDA, Applications      Week 9, Lec 1

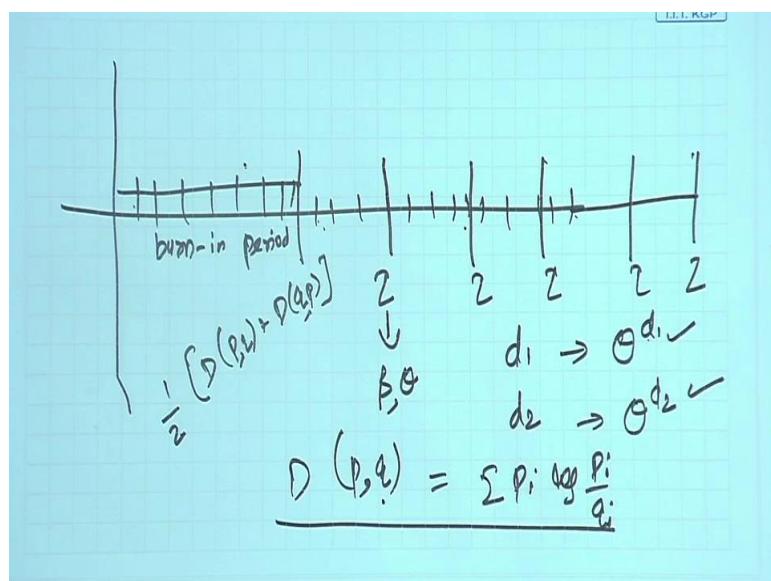
So, this is the whole algorithm in a nutshell. So, is how do you start. So, each word token is assigned to a predefined so random topic, so a random topic in 1 to T. So, you have this whole collection and you assign some random topic to each word. Now, you compute your two matrices C D T and C W T from that assignment. Now for each word token, in each iteration, what you will do for each word token you sample a new topic as per this distribution. And we have seen that once you have the matrices formulated you can find out this probability distribution and you can sample a topic by sampling from a multinomial distribution. And when you will sample a topic and put that topic in that, for that word accordingly adjust your two matrices.

So, now then you make complete single pass through all your words in your corpus that is called one Gibb sample. So, this is your one Gibb sample, and then you will do it again and again and again. So, what happens is that initially for certain iterations, you can call it as a burnin period, initially burnin period. So, where you will not store those samples, you will use

them to update the values, but you will not store, but after some word burnin period you will start storing these values.

Now, you will not store every conjugative value. So, what might happen because you are just using a previous values to compute the next one, they may be very, very correlated. So, you will have some regularly spaced at some regularly expressed interval, you will store these samples.

(Refer Slide Time: 17:57)



So, something like; so, you are doing these iterations over the full corpus. So, there will be some initial burnin period. So, you are computing Gibb sample. After burnin in period you will have reliable Gibb sample, but what will happen those that are very, very close, they will be highly correlated. So, you will say I will store some regularly space intervals. So, say I will store it up every 100, after every 100 iteration, I will store this. So, we will have cube multiple Gibb samples now. So, each sample contains all your z, all your z, all your assignment is contains. And then and from here you can compute your beta and theta. And then finally, you can take an expectation over all these values. And this will give your one particular approximation of your parameters. If we take a expectation over various samples that you are getting from Gibb sample.

(Refer Slide Time: 19:09)

### Estimating $\theta$ and $\beta$

$$\beta_i^{(j)} = \frac{C_{ij}^{WT} + \eta}{\sum_{k=1}^W C_{kj}^{WT} + W\eta}$$
$$\theta_j^{(d)} = \frac{C_{dj}^{DT} + \alpha}{\sum_{k=1}^T C_{dk}^{DT} + T\alpha}$$

*These values correspond to predictive distributions of*

- sampling a new token of word  $i$  from topic  $j$ , and
- sampling a new token in document  $d$  from topic  $j$

Pawan Goyal (IIT Kharagpur)      Gibbs Sampling for LDA, Applications      Week 9, Lecture 1



$$\beta_i(j) = (C_{ij}WT + \eta) / \left( \sum_{k=1}^W C_{kj}WT + W\eta \right)$$

$$\theta_j(d) = (C_{dj}DT + \alpha) / \left( \sum_{k=1}^T C_{dk}DT + T\alpha \right)$$

So, once you have this  $Z$ , how do you compute your betas and theta? That is very easy; we were actually using that to compute  $Z$ . So, betas are the probability that a topic  $j$  is assigned to the  $i$ th word. So, this will be from the matrix  $C$   $W$   $T$ . So, I will take  $C_{i,j}W T$  plus  $\eta$  divide over all words  $C_{k,j}W T$  plus  $W\eta$ . Similarly, what is  $\theta_j(d)$  that is what is probability of topic  $j$  under document  $d$ ; it will be  $C_{d,j}D T$  plus  $\alpha$  divide by summation over all topics  $C_{d,k}D T$  plus  $T\alpha$ . So, once we have the  $Z$ , then we can compute here beta and theta also.

So, these values will correspond to the distribution of sampling a new token of word  $i$  from topic  $j$ ; and sampling a new token in document  $d$  from topic  $j$ .

(Refer Slide Time: 20:02)

*An Example*

The algorithm can be illustrated by generating artificial data from a known topic model and applying the algorithm to check whether it is able to infer the original generative structure.

*Example*

- Let topic 1 give equal probability to MONEY, LOAN, BANK and topic 2 give equal probability to words RIVER, STREAM, and BANK

$$\beta_{MONEY}^{(1)} = \beta_{LOAN}^{(1)} = \beta_{BANK}^{(1)} = 1/3$$
$$\beta_{RIVER}^{(2)} = \beta_{STREAM}^{(2)} = \beta_{BANK}^{(2)} = 1/3$$

- We generate 16 documents by arbitrarily mixing two topics.

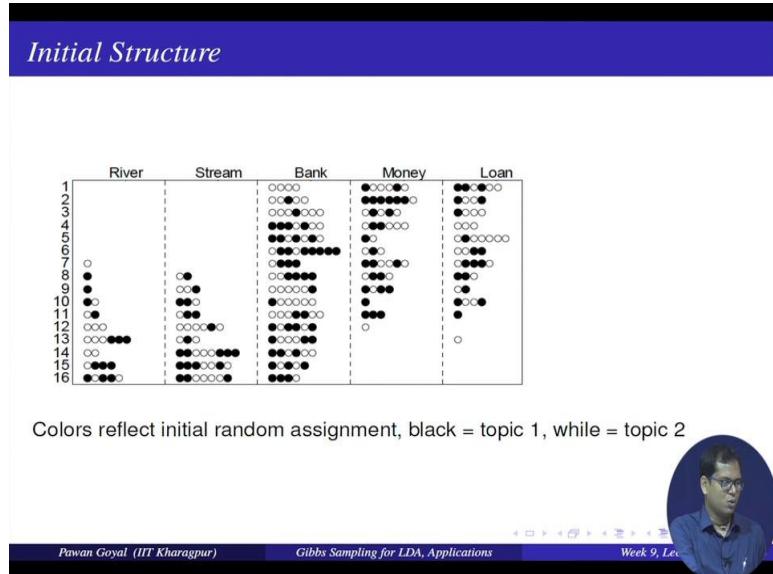


Pawan Goyal (IIT Kharagpur) Gibbs Sampling for LDA, Applications Week 9, Lec 1

Now, it is just an example to explain what it means to use Gibb sampling. So, what is an example? So, this is like So, we are taking in a artificial data and for a known topic model and applying the algorithm, we will check if we can come back to the same topic distribution that we started with. So, what is done here, let us say we have two topics. So, we are doing a generation now, and then we will see whether Gibbs sampling can infer back the original topic distributions.

So, what is done in generative part let us say I have two topics - topic 1, topic 2. And simply we are saying topic one assigns equal probability to three words money, loan, and bank; and topic two assigns equal probability word river, steam and bank. So, all these three are assigned the probability of 1 by 3 each. So, these are topic distributions.

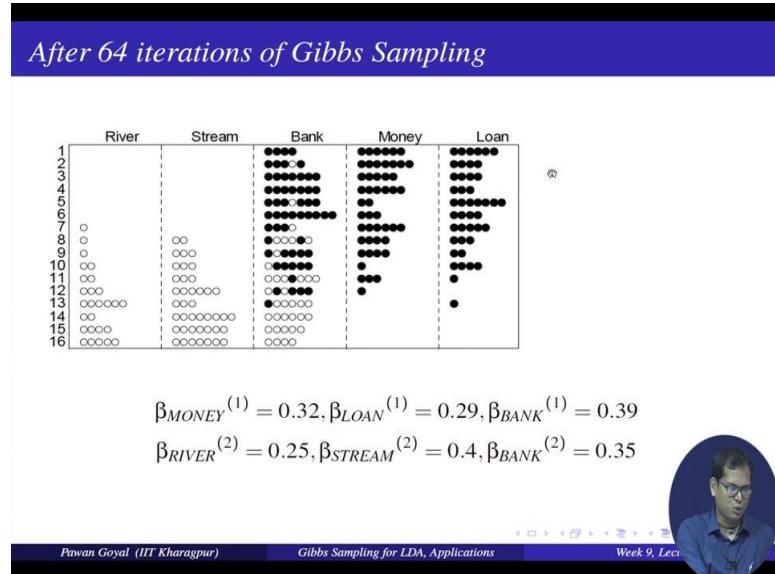
(Refer Slide Time: 21:10)



Now, by arbitrary mixture of these two topics we are generating 16 documents, so that is how these 16 documents look like. So, black means topic 1, and white means topic 2. So, there are two topics. So, each document has some number of words from different topics. So, I am sorry you should initial look at only the number of balls that is how many words are there in the document. Now, to apply Gibb sampling, so that is why you generated all these documents. So, this makes sense that you are generating documents that are having bank, money, loan lot of documents; then some document that contain only river, stream, bank. So, these are documents from only one topic these are document only from another topic, and there are some documents that are mix in these two topics, so that is how you are doing generating 16 documents.

Now, your task is can you use Gibbs sampling to find out what are the two topic distributions here. So, what is done for that, initialize all the words to some topic; so randomly that is why you are seeing the random assignments. So, black is topic 1 and white is topic 2 - some random assignment. From this random assignment, you can have the two matrices  $C_{WT}$ ,  $C_{DT}$ . And then what you will do in each iteration, you will go to each word find out what is the probability that this word will be assigned to topics  $j$  sample from the distribution update your matrices and you get some Gibb samples.

(Refer Slide Time: 22:36)



So, once you do that, so this is what you see after 64 iterations of Gibbs sampling. Can you see that? It actually looks very, very close to what we had initially. So, all these words are assigned to topic 1 and all these words are assigned to the next topic. And you can see that bank, money, loan are been assigned the same topic in a given document and that makes lot of sense. And from this particular sample, you can also compute your betas. And if you compute the betas, they come out to be very, very close to what you started with. So, started with each word having a probability of 1 by 3, and that is what is roughly what we obtain.

And this is just an explanation that how Gibbs sampling can help you to recover what is a original topic distribution this is from artificial data. But now you can do that for any real corpus, we have the real corpus and we want to find out what is the topic distributions apply Gibbs sampling and find out that. So, there are various tool kits available that I also disused in the last lecture. So, where you can give a corpus, you can define the number of topics and they can give you the all these values what are theta, what are betas, what are per topic per document per word topic distribution.

(Refer Slide Time: 24:00)

**Computing Similarities**

**Document Similarity**

Similarity between documents  $d_1$  and  $d_2$  can be measured by the similarity between their topic distributions  $\theta^{(d_1)}$  and  $\theta^{(d_2)}$

KL divergence :  $D(p, q) = \sum_{j=1}^T p_j \log_2 \frac{p_j}{q_j}$

Symmetrized KL divergence:  $\frac{1}{2}[D(p, q) + D(q, p)]$  seems to work well

**Similarity with respect to query  $q$**

Maximize the conditional probability of query given the document:

$$p(q|d_i) = \prod_{w_k \in q} p(w_k|d_i)$$
$$= \prod_{w_k \in q} \sum_{j=1}^T P(w_k|z=j)P(z=j|d_i)$$

Pawan Goyal (IIT Kharagpur) Gibbs Sampling for LDA, Applications Week 9, Lec1

$$D(p, q) = \sum_{j=1}^T p_j \log_2 (p_j/q_j)$$

Now, once you have that what are sort of simple tasks that you can do with this. So, for example, one very important task is can you compute similarity between two documents and how will you do that. So, suppose I have two documents given in  $d_1$  and  $d_2$ . To compute the similarity between that I will see what is the topic distributions for  $d_1$  and  $d_2$ . So, if they are similar, I will say that the two documents are similar, but if their topic distributions are different, I will say they are different plus. So, I have two documents  $d_1$  and  $d_2$ , I find out what is theta for  $d_1$ , what is theta for  $d_2$ .

Now, I can say distance between two distribution  $p, q$ , I can use the KL divergence summation  $p_i \log p_i / q_i$ . So, now, I compute the KL divergence between the two topic distributions for  $d_1$  and  $d_2$  and that will get me what is the distance between the two documents that is one every standard measure for finding out how similar two documents are. Because this is asymmetric, you can also do something like  $1/2 [D(p, q) + D(q, p)]$ . So, we can also use that as some sort of distance symmetric, so that is you have now the corpus you find know the topic distributions. Now, you can use that to compare the similarity between any pair of documents, so that is one very, very important application of topic models.

Then you can find out what is similarity of the document with (Refer Time: 25:46) query, what is the probability that query is generated from a document that is what we study in information

(Refer Time: 25:53). That you will have a lot of documents, you want to find out given a query which document should be ranked higher this will be computed using what is the probability that the query is generated from this document. So, whichever documents give the highest probability of generating the query is given the highest score. Now, what is the formulation you want to find out probability of query given the document? The query is nothing but a set of words.

(Refer Slide Time: 26:22)

$$\begin{aligned}
 p(q|d_i) &= \prod_{w_k \in q} p(w_k|d_i) \\
 &= \prod_{w_k \in q} \sum_{j=1}^J \frac{p(w_k|z=j)\beta}{\theta_j}
 \end{aligned}$$


So, if we take it simply easily multiplication over all the words in my query probability  $w_k$  given  $d_i$ . Now, how do I compute probability of word given  $d_i$  there I am using the LDA, the LDA model. So, I will say, so I will marginalize it over all the topics. So, this will be covered all words in query summation over all topics probability  $w_k$  given topic is  $j$  given document  $d_i$  this is nothing that comes from your beta directly that comes here from your theta directly. And use that to find out what is the probability of this query given this document. So, this is again a very interesting use.

(Refer Slide Time: 27:22)

*Computing Similarities*

*Similarity between two words*

Having observed a single word in a new context, what are the other words that might appear in the same context, based on the topic interpretation for the observed word?

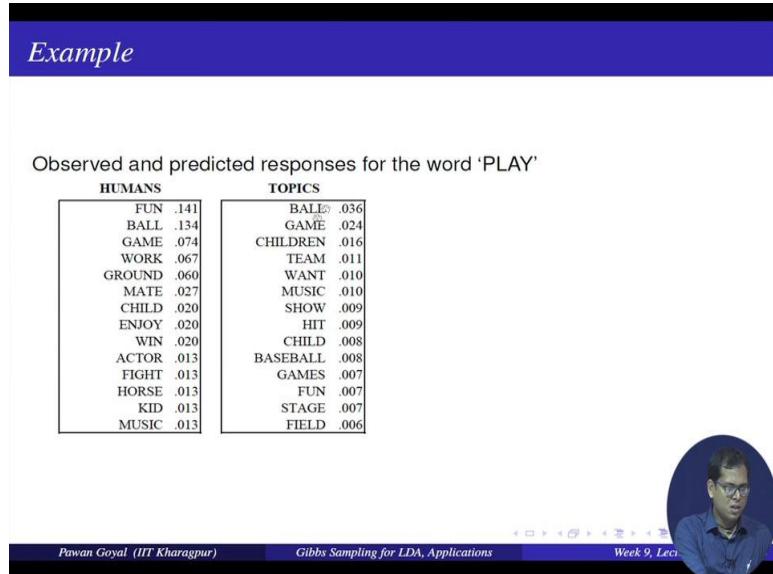
$$p(w_2|w_1) = \sum_{j=1}^T p(w_2|z=j)p(z=j|w_1)$$

Pawan Goyal (IIT Kharagpur)    Gibbs Sampling for LDA, Applications    Week 9, Lecture 1

$$p(w_2|w_1) = \sum_{j=1}^T (p(w_2|z=j)p(z=j|w_1))$$

Then you can also use it to find out which two words are similar, what is the probability of word w 2 given w 1 that is nothing but again you marginalize over all the topics, and this you will compute from either your beta or by using Bayesian theorem. So, this will give me given a word w 1 what are some of the likely words in my vocabulary. And this you can find from computing words similarity and all that you have to using doing using distributional similarity and other stuff. So, that is why again a nice method for capturing semantics between words, documents, even sentences.

(Refer Slide Time: 28:02)



So, how do you validate that this works, so this is very simple experiment. So, you have the word play. By using topic model, you find out which words have the highest probability given the word play. So, probability of  $x$  given play, find out some top words. Then you ask some humans to say that when you hear the word play what words come into your mind that is humans. So, words like fun, ball, game, work, ground, mate, child etcetera they come to their mind. Then you try to see whether these two lists are similar. And you find that many words are similar like ball, game they come on top even in topic model. So, this is the interesting way of evaluating whether a model is doing well for capturing words in that.

So, we discussed how do we use Gibbs sampling to estimate these parameters and some simple applications. Next, we will also talk about some non-parametric Bayesian model and what are the different applications they can be used for.

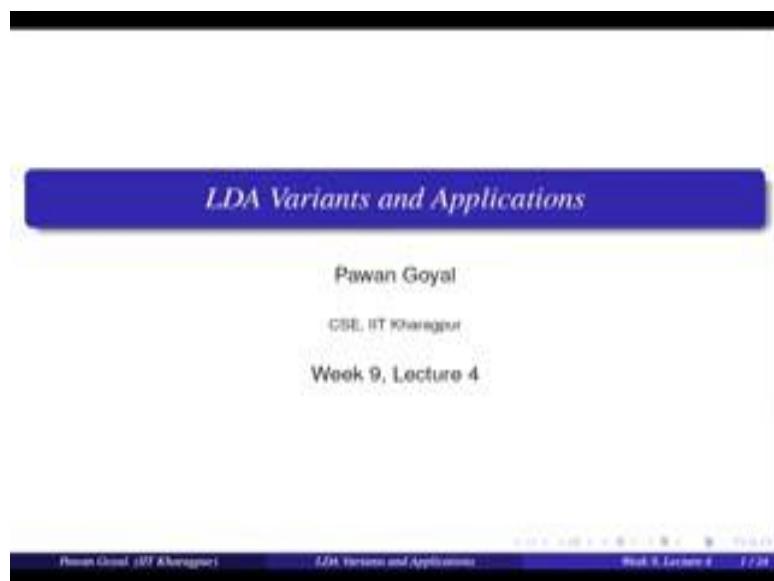
Thank you.

**National Language Processing**  
Prof. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 45**  
**LDA Variants and Application – I**

So welcome back for the fourth lecture of this week. So we have been talking about topic models.

(Refer Slide Time: 00:24)



And in the last lecture we had covered what is generative model of LDA. And how do you Gibbs sampling to estimate the parameters of LDA by using the observations as a various documents and whatever works regarding the documents. So in this lecture in the next we will be talking about different variants of LDA and how do we use that for different applications. So you may not cover many of these topics in detail, but once you get the idea of what these variants are given application you can go and look more into these topics.

So this lecture I will be starting with some sort of classes size on Gibbs sampling that how do you estimate parameters from a given Gibbs sample and then we will go forward for the variants of LDA.

(Refer Slide Time: 01:10)

**Example Problem**

Suppose you are using Gibbs sampling to estimate the distributions,  $\theta$  and  $\beta$  for topic models. The underlying corpus has 5 documents and 5 words, {River, Stream, Bank, Money, Loan} and the number of topics is 2. At certain point, the structure of the documents looks like the following Table. For instance, the first row indicates that the document 1 contains 4 instances of word 'Bank', 6 instances of word 'Money' and 6 instances of word 'Loan'. Black and white circles denote whether the word is currently assigned to topics  $t_1$  and  $t_2$  respectively.

Use this structure to estimate  $\beta_{MONEY}^{(2)}$  and  $\beta_{BANK}^{(1)}$  at this point. You can take the values of  $\eta$  and  $\alpha$  to be 0.1 each.

Doc. Id	River	Stream	Bank	Money	Loan
1			●●●●	●●●●●●	●●●●●●
2			●●●●●●	●●●●●●●●	●●●●●●
3	○	○○○	●○○○●○	●●●●●●	●●●●●●
4	○○○○○○	○○○	●○○○○○	●●●●●●	●●●●●●
5	○○	○○○○○○	○○○○○○	●●●●●●	●●●●●●

Praveen Chalapathy IIT Kharagpur IIT, IISc, and Applications Week 8, Lecture 4 3 / 34

So let us take this example problem. So what we are given here. So you are saying that there is a corpus that has 5 documents, so 1 2 3 4 5 a document id is here. And 5 words river stream bank money and loan. And there are only 2 topics that you want to estimate. Now this is when you are doing Gibbs sampling at certain point of time you are given what are the different assignments of topics to different words in the document. So what do you see here document one has 4 words bank 1 2 3 4 5 6, 6 times money and 6 times loan and at that point of time all these 16 words have been assigned to topic 1.

So black is topic t 1 document 2 has most of the words assigned to topic t 1 and one word to topic t 2 and so on. You are given the topic of assignment at a given point of time. So you can see that the first show indicates that the document 1 contains 4 words for instances of the word bank 6 of word money and 6 of word loan and black and white circles are topic t 1 and t 2.

Now, your task is that you want to use the system share to estimate different parameters of your mountain. So remember what are the 2 main parameters, one was your theta another virtual beta. Beta is what is the probability of a word given a topic and theta is what is the probability of a topic given this document. So in this example we will try to estimate 2 different beta values. That is what is beta money to probability of money in topic 2 and beta bank 1 probability of bank in topic t 1 and you are given that eta and alpha are 0.1.

Now, if you remember the formula how do you compute beta money to for that you will need to use your matrices.

(Refer Slide Time: 03:10)

$$\beta_{\text{MONEY},i} = \frac{C_{ij}^{WT} + \eta}{\sum_k C_{kj}^{WT} + W\eta}$$

$$C^{WT} \begin{matrix} \text{River} \\ \text{Stream} \\ \text{Bank} \\ \text{Money} \\ \text{Loan} \end{matrix} \begin{bmatrix} 0 & 12 \\ 0 & 9 \\ 11 & 16 \\ 17 & 2 \\ 13 & 0 \end{bmatrix} \quad \begin{matrix} C^{DT} \\ t_1 \\ t_2 \end{matrix} \quad \frac{0 + 0.1}{37 + 0.5} = \frac{0.1}{37.5}$$

$$\beta_{\text{BANK}} = \frac{11 + 0.1}{41 + 0.5} = \frac{11.1}{41.5}$$

Remember you consider 2 matrices  $C^{WT}$  and  $C^{DT}$ .  $C^{WT}$  this word assigned to what topic  $dt$ . This document what are topics that are sent. For this problem for beta money integrally what is that, what is the probability of the word money for the topic -  $t_2$ . So we will need only this matrix this word assigned to what topic in terms of this matrix. How do you write this for 2 parameters you say  $C^{WT}$ ? So if you have to write  $ij$  ith word  $j$ th topic. So let us say this is my  $i$  and this is my  $j$ . So this is  $ij$  plus you have used the hyper parameter  $\eta$  divided by now you see all the different words that are assigned to this topic. Summation over  $k$   $C^{WT}$ ,  $w_j$  sorry it should be  $k_j$  for all  $k$  plus  $W$  times  $\eta$ . So this summation  $k$  for all  $w$  and that is why you have  $W\eta$  here. That is why you estimate this parameter beta money too.

So let us see how do we estimate this parameter from this matrix. So one thing is that you will have to first construct this matrix, so let us see what does this matrix look like  $WT$  will have 5 words right. River, stream, bank, money and loan and you have to find out how many times this word has been assigned to topic  $t_1$  and  $t_2$  and not including this instance fine, so that we cannot do at this time. So we will take all the instances. So let us see. River is not assigned to only topic  $t_1$  black. So river sorry river is not assigned to topic  $t_1$  at all only the topic  $t_2$ . So I have 1 2 3 4, 1 2 3 4 5 6 7 8 9 rivers is send to topic

to 9 times. A string again 3 3 6 and 6 12 0 12 bank 1 2 3 4 5 6 7 8 9 10 11 11 times topic t 1 and 1 2 3 4 5 6 7 8 9 10 11 12 13 14 to 16 topic t 2, money only topic t1 1 2 3 4 5 6 1 2 3 4 5 6 7 13 and for 17 and loan 1 2 3 6 10 13 that is your matrix C wt.

Now, let us see how do we come to beta money, for 2 you have to compute C wt ij, i is money yes and j is t 2. So this is 0 it is not a second topic t to any number of times I write 0 plus eta h 0.1. Now divided by summation over k all the words, any all the words that when you have been assigned to topic to here. So I will just add this column 9 plus 12 plus 16. So that will give me 37 plus w number of words is 5 times eta this ones 0.5. So this comes out to be 0.1 divided by 37.5 similarly can I come to beta bank 1 for that I will find out how many times bank has been assigned to topic t 1 11 plus eta 0.1 divided by summation over k c kj wT. So that will be how many words that have been assigned to topic t 1. So it will be 11 plus 17 20 8 plus 13 41. So 41 plus 0.5, so this comes out be 1.1 divided by 41.5.

So like that you can compute all there your different betas at this given time point. So well you can compute here thetas. So this can this you can take as an exercise find out what is theta for document 1 or 2 for different topics. This is something that you can do.

Now one more thing that that might be interesting, suppose I ask you question that find out in this iteration what is the topic that will be assigned or what will be the multinomial distribution from which you will sample a topic for a given work. Like the first bank in this document. First they are for instance is your bank for the first national bank you have to assign a new topic. That is what you doing iterations. So for that you will have to again compute different betas and thetas, but what you have to keep in mind you have to exclude the current instance. So when you are computing this you will remove the current instance.

So suppose this is 11. So you are removed then one from here and so on. You will compute each values and from by removing the conditions you will compute the betas and thetas and use your formula for find out what is the probability for topic t 1 quality to from this distribution you will sample a topic. There is something that you would keep in mind. So this was a simple example for how do you use Gibbs sampling to estimate your parameters.

Now, we talked about certain applications of LDA in the last lecture. So we saw that we can use it for computing similarity between words to complete similarity between documents that are one of some of them very promising applications, but what are some other different tasks where you can use these topic models for.

First let us see the simplest task. That is can we model the documents using the topics that is the straightforward thing that you can do using this LDA.

(Refer Slide Time: 09:39)

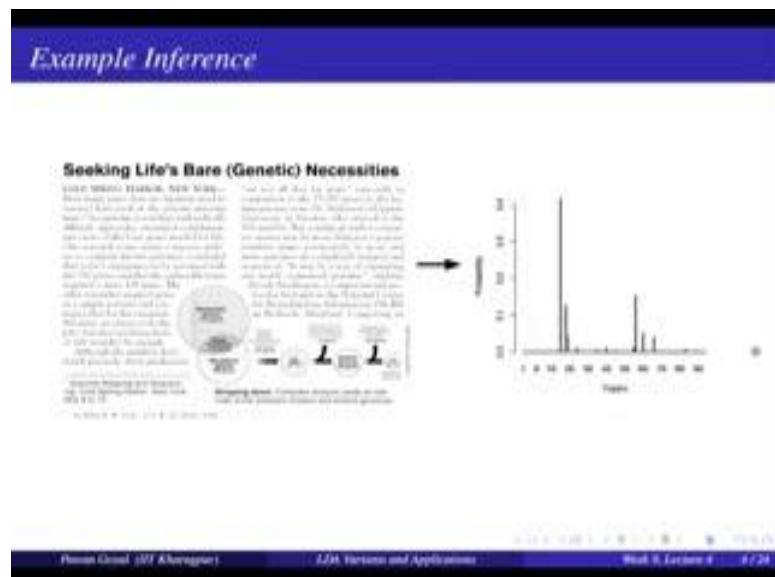
*Data*  
The OCR'ed collection of Science from 1990-2000  
• 17K documents  
• 11M words  
• 20K unique terms (stop words and rare words removed)

*Model*  
100-topic model using variational inference

So here is something the collection that was also one of the motivation with which we started these topic models. So we are taking collection of science papers from 1990 to 2000. So there are 17000 documents and 11 million words there and there are 20k unit terms 20000 different terms after removing the stock words and real words. Now on this collection suppose you run your LDA model. So for running the LDA model you need to tell what is the number of topics.

Suppose you see it 100 topics. So once you have earned your model using your 100 topic models and you can use either Gibbs sampling or variational inference. So these are 2 different possibilities for estimating your parameters. Now once you have done that try to see what are what do your documents look like, what are the topic distributions there.

(Refer Slide Time: 10:30)



So when we do that, remember this was the article that we were looking at seeking life's bare necessities and this for genetic in the parentheses. So we found 3 4 topics there right, compositional some data analysis some genetics evolutionary biology and so on.

Now, suppose we run this topic model over this whole corpus we find out what happens to this document. So this document gets a probability assignment like that. So there are hundred topics and some topics get high probability. So they are few topics that are getting high probability. And then we go back and look at these 4 topics what are the most common words in these 4 topics. So we see something that we were looking for.

(Refer Slide Time: 11:08)

Example Topics			
human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	now	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

So we saw from first topic contains words like human genome dna genetic. So it is about genetics. Then the second topic is evolution in biology third about different disease and bacteria and forth about the data analysis. So these are the 4 topics that come on top.

And this looks very interesting that from by you do not give any information to this model that this document contains these topics or which document contains these topics, is still by learning from a large corpus it was able to learn different topics and the topic assignment for a given document. So this is very interesting aspect of LDA.

(Refer Slide Time: 11:46)

Modeling Richer Assumptions in Topic Models			
• Correlated topic models	• Dynamic topic models	• Measuring scholarly impact	

So, now apart from modeling simple topics that are there in the document what else can be modeled using these topic models. So we will see how do we model different other junctions in the data it. So till now what we are saying. So we have a static data. So we have a static data or whatever time it spends. So there is fix set of topics. And the topics are also kind of independent of each other. You do not say if in the document our  $t_1$  occurs then  $t_2$  should also occur we do not say that, but can we also model these assumptions. So for that we have different model models like correlated topic models dynamic topic models and measure measuring scholarly impact.

(Refer Slide Time: 12:33)

**Correlated Topic Models**

- The Dirichlet is an exponential family distribution on the simplex, positive vectors that sum to one
- However, the near independence of components makes it a poor choice for modeling topic proportions
- An article about fossil fuels is more likely to also be about geology than about genetics

**Using logistic normal distribution**

A multivariate normal distribution of a  $k$ -dimensional vector  $x = [X_1, X_2, \dots, X_k]$  can be written as

$$x \sim N_k(\mu, \Sigma)$$

with  $k$ -dimensional mean vector  $\mu$  and  $k \times k$  covariance matrix  $\Sigma$

Praveen Choudhary (IIT Kharagpur) LDA: Versions and Applications Prof. S. Lakshmi

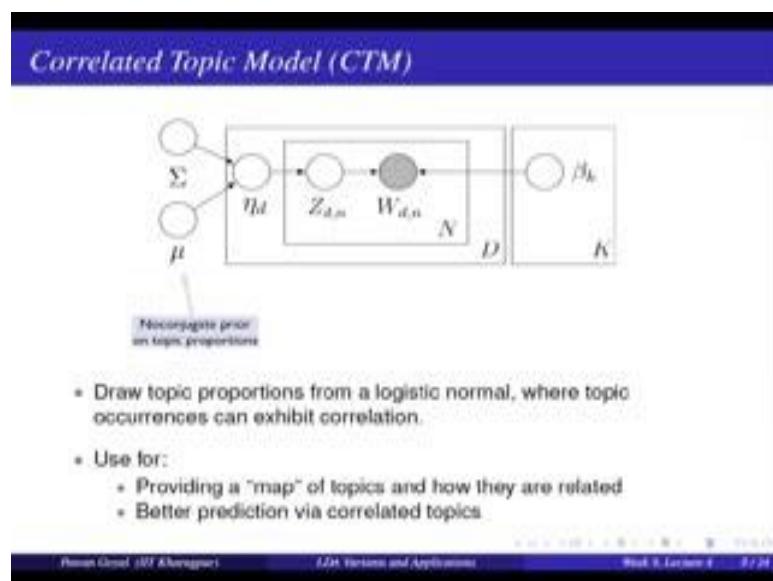
So we will see how do we go from LDA to any of this variance. So let us see the correlated topic models. So right now what we are doing? You are having additional distribution that helps me sample the probability distribution of topics for a given document. So this is what this is some simplex where there are positive vectors not in the probability that I add up to one; however, in this digital distribution the components of the probability distribution are quite independent of each other so; that means, they do not model various dependence between the topics.

So suppose I want to say that these are article about fossil fuels. And if I know the topic fossil fuels occurs in the article probably the topic about geology may also occur rather than genetics. There is something that I might know that these 2 topics are quite correlated and these 2 topics are not correlated. So can I use this intuition to battle on my

topics and distributions within the documents that certain topics are co related they will occur together certain topics are not correlated they will probably not occur together. So this cannot be modeled by using the distribution vision. So we use a different distribution to model the topics in in a document and that is where we use the multivariate normal distribution.

So something like this. So you are having  $k$  topics. So you will have a multivariate normal distribution, where you are having, your sampling this  $k$  dimensional distribution, but now from this normal distribution with a mean and covariance. So mean will be what is the private information that we have about different topics, what will be the mean of the different topics and sigma will be how are these different topics co related with each other. That is what you will try to give us in your model.

(Refer Slide Time: 14:25)

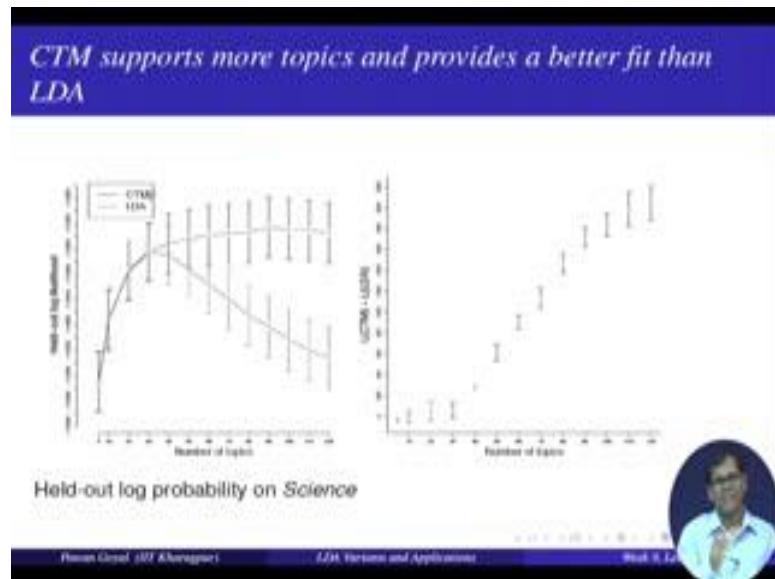


So how does your model change? So everything remains the same, except that instead of sampling from additional distribution you are now starting a sample from a multivariate logistic normal distribution with a mu and sigma. So eta these are samples from this distribution so that is where the topics can exhibit various correlations, that these 2 topics are correlate with each other while these 2 are not correlated.

So once you have done that finally, what you will get you will again get your  $k$  topics right like hundred topics you are doing in the case of science, plus you will also know which 2 topics or which pair of topics are correlated with each other. And this can be

very nicely used to give a map that these topics are make a single cluster a single group they are correlated to each other, these topics are against other no group that are correlated with each other.

(Refer Slide Time: 15:18)



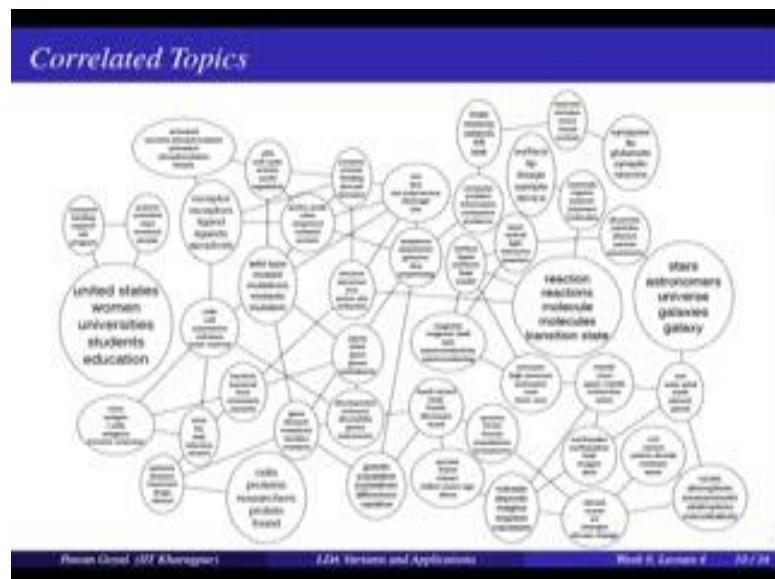
So for example, if it is and how do you know that this works better than LDA. So one good method of evaluation is that you try to find out what is the log likelihood that this providing to a held out data held out data is some data that you did not use for training of your our topic model. So you do not give it as an input for Gibbs sampling and or variational inference. So once you have learnt the topics, try to see what probability it gets to a held out data some separate data again from a same domain. So whatever topic model gives you a better likelihood that is probably better that is a better model. This is similar to what we did in the case of language modeling.

We found out what is the perplexity that it assigns to a held out data. Similarly, here what is the log likelihood that is essentially different held out data. So we see if the colder top model gives a better likelihood then the LDA simple LDA and that is what you see here. So this is on the held out likelihood you want a number of topics. So interestingly if the number of public is small say 30 to 40 both models give the same log likelihood, but as you increase the number of topics the LDA model the likelihood given LDA model is starts decreasing, but this does not happen with CTM model; that means,

if you want to have more topics then CTM is a better choice than LDA. CTM can model reach your assumption in the data then LDA if you have more number of topics.

So that you can see here also likelihood difference between CTM and LDA it keeps on increasing as you increase number of topics.

(Refer Slide Time: 17:01)



So this is how the map will look like. So you will see here this topic talks about United States women universities students in education and it is about research funding support sciences scientists and research people. So these are correlated topics, but they are not so correlated to the topics like here a stars astronomers universe galaxies and galaxy. So which again make different sort of clusters for topics and this which model simply by using the covariance matrix. These topics are connected together now. So this was one assumption that we can model.

(Refer Slide Time: 17:33)

*LDA assumption*

- LDA assumes that the order of documents does not matter
- Not appropriate for corpora that spans hundreds of years
- We might want to track how language changes over time

Now, suppose just take another assumption. That is which topics are correlated to each other and which topic, sorry how do topics change over time. So right now what we are assuming, so you have a static corpus and in which there are the same topics over time the same set of topics over time working. And by topic I mean the distribution of words in the topic are also same over time, but this is not true in general. Suppose you have a collection that it spends on multiple decades or even centuries say 200 years of data.

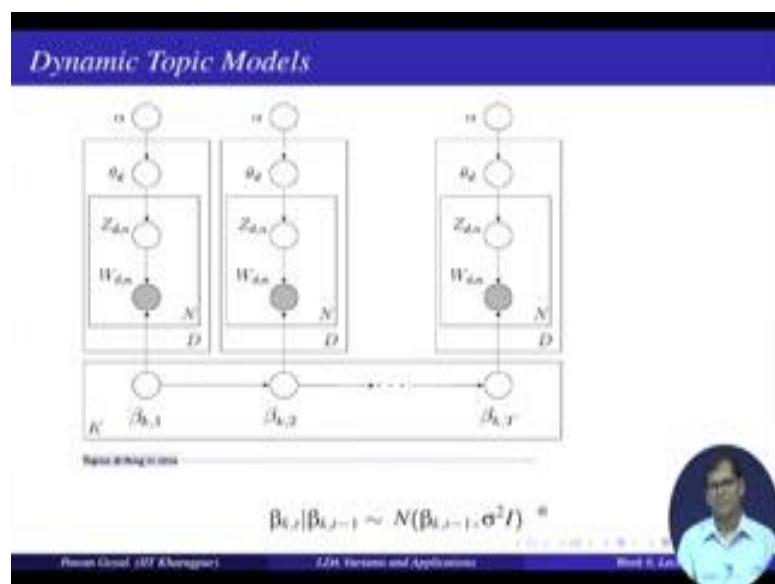
So what you will see? The same topic the number of work, the type of words that you are seeing over the time are changing initially you will see some different sort of words and later on you will see some different sort of words topic might be the same, but then the kind of words will keep on changing. Also the probabilities of words will keep on changing. Now this you cannot model by a simple LDA model. So how do you actually specify this and that is where a dynamic topic model is used?

So what is the problem with LDA? So it assumes that the order of document does not matter and this is not appropriate for the corpus that are spending for hundreds of years. So we might want to track how the language within the topics are changing over time and for that we use dynamic topic models. So it is very interesting this is just diet extension of LDA, but now when you model that how the topics are changing over time. So how do you do that? So when you have a large collection we will divide you into

multiple different time points. So you say this is your corpus 1 corpus 2 corpus 3 corpus 4 and so on.

Over time you are starting from one up to the last corpus. Now when you see, when you define your topic distributions you say that let us say the initial corpus had distribution beta k 1 beta k for time step 1. So what you will say as you go from time stamp 1 to time stamp 2 the next beta will not be the same as the beta 2 will not be the same as beta 1, but will be again a distribution is starting from with the mean of beta k 1 with some variance. So that is you are allowing to change the probabilities of words within the topic model. And that you can do over time. So that is the previous topic topics influence the next topics, but the next topics can also change with certain variance and this is how the model looks like.

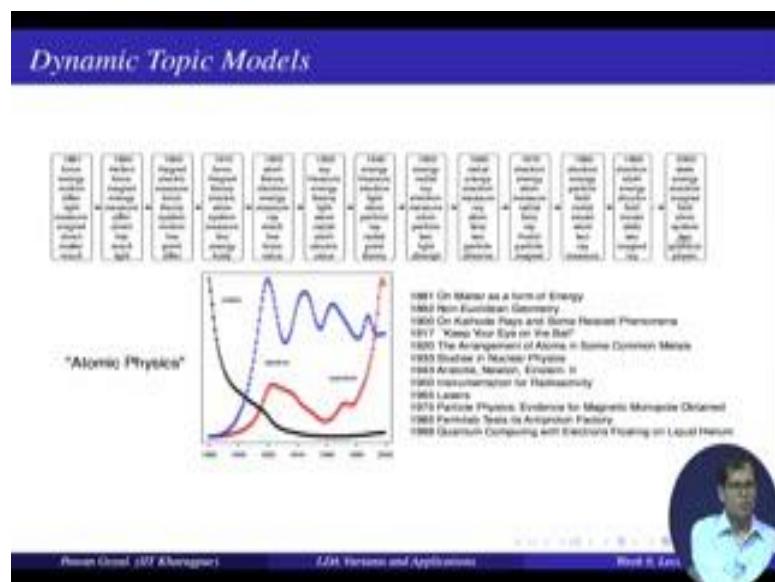
(Refer Slide Time: 20:05)



So we are having time stamps from 1 to t. So this is same as if you are having T, capital T different different copies and you are running topic model for each corpus. But now you are not doing it independently because your betas are connected. So you are saying beta k 1 is an input to beta k 2. So this is like a normal distribution beta k2 is like a normal distribution with a mean of beta k 1, but with some variance. So you are biased to take same words with same probability, but with certain variance. So that will allow changing the probability distribution of words and also having new words in the topic model.

So that is the only thing that happens. So you are having different betas over time, but they are connected it is starting from the first thing up to last time. So what you are seeing the previous time point topic models will influence the top model at the next time point. Now once you have this dynamic topic model how it can help.

(Refer Slide Time: 21:06)



So suppose you are modeling how in science a particular topic is changing over time.

So let us say this is modeling of science is starting from at 1881 to 2000 and the topic is atomic physics. So what you see the kind of words that are there in the topic keeps on changing the over time. So here you have words like force, energy, motion, differ light, major magnet, direct matter and result, but as you go over time the words here are energy electron magnet field atom system to quantum physics. So you see the word quantum comes up. And the word electron comes up that are not there in the initial time point. So this you can see also over the time and this is a nice plot that shows how this 3 words vary over the decades in this in this topic.

So see, you initially start with matter having a very high probability, but then we start decreasing over the decades the word electrons come up at certain time point this is around 1900 with this cathode rays experiment and is start, this is like a stable like on each stable over time, but then the new topic on quantum also comes up, and that is having a very high probability. So this gives you a nice visualization that within this given topic how the words are evolving over time, how is topic evolving over time.

Similarly, if you see the topic of neuroscience, you can similarly observe that initially the word nerve was having a very high probability, but over the time you get the word like neuron coming into picture and then the ca2 that is like in area, where you will the particular area ca2. And this you can also correlate with what the difference sort of papers says seminal papers that are published, that might have given rise to these terms coming up into these topics. So this was interesting that in science in the same topic how the different words keep on coming over time.

Now, this also gives rise to a nice application, that is can you model what are the most influential articles in science influential papers in science. So what will be the idea influential paper is the one that will affect the topic model. That will affect the change in the topic model. So with each document you might have an influence variable and the idea is that the change in topics are more affected by the influential vapors than the non-influential papers. And that way we can model which article is more influential in another. So how will this model look like?

(Refer Slide Time: 23:42)

*Measuring Scholarly Impact*

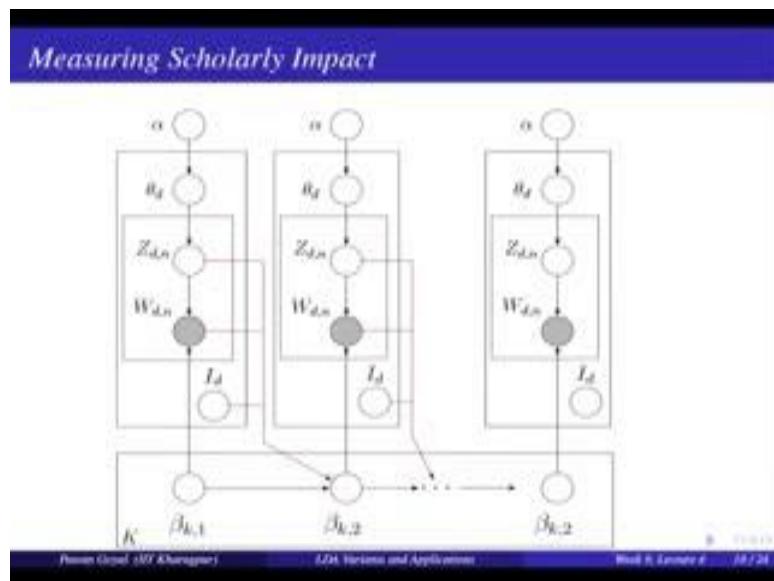
*How to model influence?*

- Idea from Dynamic Topic Models, influential articles reflect future changes in language use
- The influence of an article is a latent variable
- Influential articles affect the drift of the topics that they discuss
- The posterior gives a retrospective estimate of influential article

Praveen Choudhary (IIT Kharagpur) LDA: Variants and Applications West N. Smyth

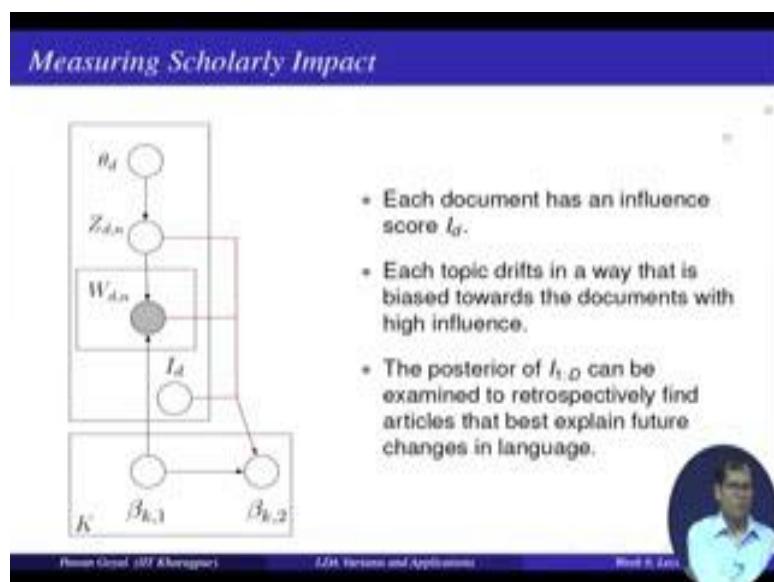
So that is influential articles reflect the future change in the language usage in the topic. And the influence of the article can be thought of as a latent variable and what we will do influential articles will affect the drift of the topics that they discuss. So that way we can model it as a variable and the posterior that will be read for this variable will tell me how influential this article was.

(Refer Slide Time: 24:12)



So again I make a very small change to the model. That is now beta k 2 instead of depending only on beta k 1 it also depends on this id in influence variable. So how influential this article was again depending on the topic of this article. So now, this id is there with each document and finally, while I compute the posterior I find out which documents get the highest influence. So which ever get document will get the high higher id will be the influential article. So this this will remain the same as the dynamical models only now it is also estimated by this id parameter.

(Refer Slide Time: 24:53)



So each document you see here has an influence score id, and each topic drifts in a way that it is biased towards the documents that are having a high influence. This is a posterior that I will estimate from the data. And you can explain the changes in the future.

(Refer Slide Time: 25:14)

The slide has a blue header bar with the title "Supervised settings of LDA". Below the header, there are two main sections:

- Use data points paired with response variables**
  - User reviews paired with a number of stars
  - Web pages paired with a number of likes
  - Documents paired with links to other documents
  - Images paired with a category
- Supervised topic models**

are topic models of documents and responses, fit to find topics predictive of the response

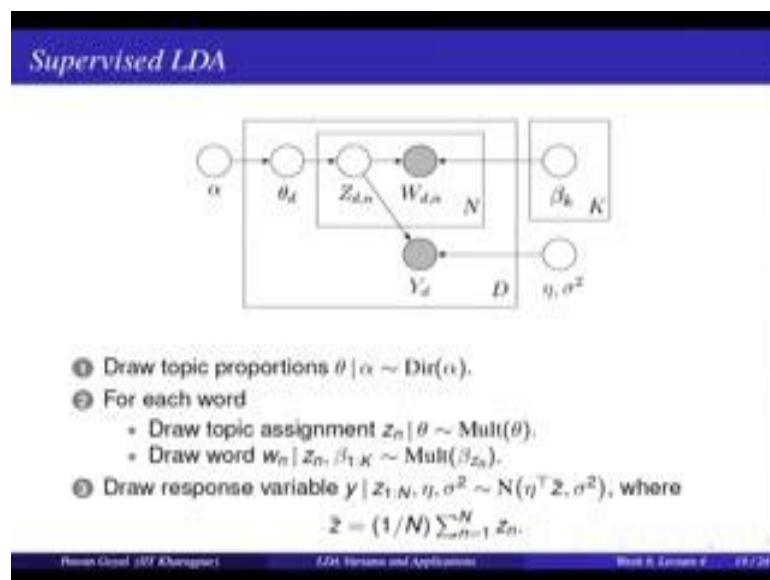
At the bottom right of the slide, there is a circular profile picture of a man with glasses and a light blue shirt. The footer of the slide contains the text "Prasen Ghosh (AVK Alorajgarh)", "LDA: Versions and Applications", and "Prof. K. Eswaran".

Now, let us see another very interesting variant of LDA, that is can be used in the supervised setting what do you mean by a supervised setting. So till now we are saying I have a set of documents in the document certain data occurs and I give it to my model. And by using Gibbs sampling evaluation influence I can find out what are different document which different topics occur in different documents that is what we can do and we can model certain assumptions like how topic change over timings, and which are correlated etcetera.

What you are saying now in the supervise settings, can we also use it to do certain prediction like think about the movie reviews with certain ratings. So can I use this model to say, this review will get that many ratings. Suppose in the text can I predict the ratings or web pages' link paired with a number of likes. How many likes this web page will get and the document pays with the link of other documents or individual with pair with the category. So lot of examples where the data points have some sort of class or a category, can you also model it using topic models. So this is where 2 different ways in which it can be done we will talk about supervised topic models and see the, what is the other variation.

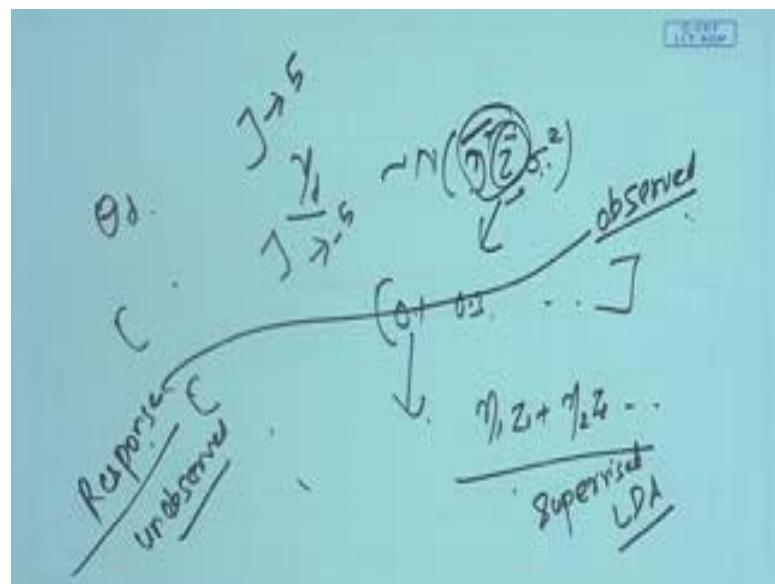
So what is the idea? So you are modeling the documents along with the responses or the categories. And the responses are those they are fitted they are fitted to find topics that are predictive of the response that is how do the topics tell about the response, so how do the topics correlate with the responses.

(Refer Slide Time: 26:56)



So how is it done? See you; this is the plate notation for the LDA model. So this is what we had seen earlier. So your beta k alpha and all that and what is additional here. So the draw topic proportional in each word what is the topic assignment plus for a given document that is your data point. So you are also finding out. So you are seeing what is the topic, distributions for the document, from there you are sampling what is your response. So from your  $Z_{dn}$  you are sampling your response that is like a normal distribution over  $\eta$  transpose  $\bar{z}$  and a variance  $\sigma^2$ . So what we are seeing this response variable depends on the topic distributions with the variance. So let us just quickly see what it means is that.

(Refer Slide Time: 27:49)



So  $Y_d$  it is sampled from a normal distribution over  $\eta$  transpose  $z$  bar and sigma square.

Now, what is  $z$  bar?  $Z$  bar as a topic proportions of this documents. So I know this topic  $t_1$  occurs 0.1 times  $t_2$  occurs 0.3 times and so on.  $\eta$  like the weights given to different topics so; however, will be like if this topic is coded with the higher response and it can be negative also if there is a negative response  $\eta$  can be negative. So these are like the weights given to different topics. So what whether this topic will go to a higher response or lower response. And sigma square is this will give you the mean and then you will sample the actual response with this  $p$  n certain variation.

So this will give you a scalar wait,  $\eta$  transpose  $z$  plane will give you a scalar  $\eta_1 z_1$  plus  $\eta_2 z_2$  and so on. This will be a scalar. So this will be a link and with this way with this variance you will sample your response. So what you need to do? You need to find out what are your  $z$  prime and you need to estimate what are your  $\eta$ s. So you have to estimate both  $z$  prime and  $\eta$ s from your model.

(Refer Slide Time: 29:05)

**Supervised LDA: why a different model is required?**

*Think of an alternative approach using original settings of LDA*  
Formulate a model in which the response variable  $y$  is regressed on topic proportions  $\theta$

*Why then a different model?*

- The response variable can be treated as an important observation to infer the topic probabilities in a supervised manner



Prof. Dinesh Manocha  
IIT Kharagpur  
IIT, Kharagpur and Applications  
Prof. D. Manocha

So now we were saying there are there is also an alternative to using the supervising LDA. So you can say why do we here, why are we doing taking this complicated model where we are having to estimate eta or using the model. Why cannot we run our topic model get my theta for each document theta d; that means, again we have a distribution over topics and then then there are something like a regression. So how does this theta give me certain is course - if liking of 5 another theta gives me minus 5 and so on that is another possibility. So I have different thetas for different documents and I run a regression from this theta to actually score. And this is called LDA plus regression model and what we are doing right now is a supervised LDA, where we are sampling this inside my model during the estimation time.

So what will happen here in this case, you are not using the response as an observed variable it is remains unobserved. In this case where wherever whereas, here response is also observed, so what is the intuition is it? If you take your response also as an observed variable then your topics can be much more aligned to the actual responses whereas, here the responses are not aligned to the topic topics are not aligned to the actual responses, and you have to later fit on later find out a mapping from the topics to the actual response. So this can be done here in the model itself that is why supervised LDA works better for taking rating the responses then in LDA model plus regression. So this is where you are taking the topic proportions theta and building your regression to find out

y another model. So here response variable can be taken as an important observation to infer the topics in this lowest manner.

So that is the interesting thing here.

(Refer Slide Time: 31:19)

*Prediction*

- Fit sLDA parameters to documents and responses. This gives:
  - topics  $\beta_{1:K}$
  - coefficients  $\eta_{1:K}$
- We have a new document  $w_{1:N}$  with unknown response value.
- We predict  $y$  using the SLDA expected value:

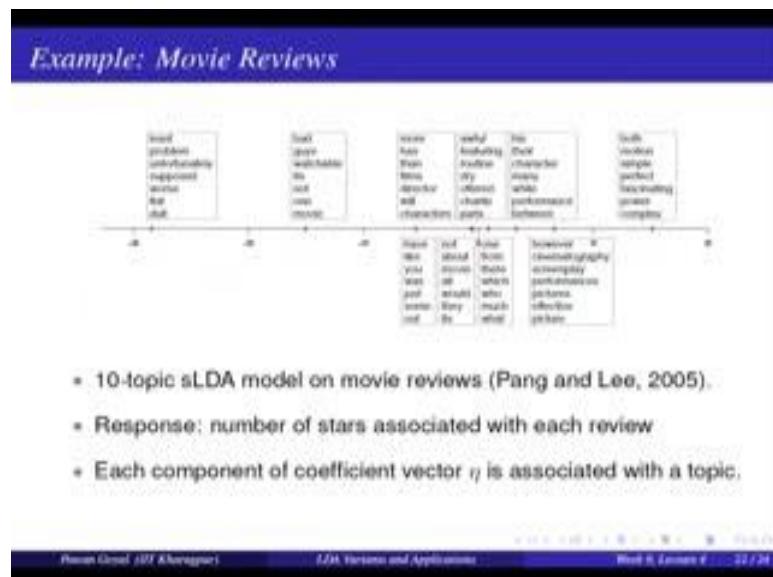
$$E[Y | w_{1:N}, \alpha, \beta_{1:K}, \eta, \sigma^2] = \eta^T E[Z | w_{1:N}]$$

Power Point, MIT Khurramjeev  
LDA, Versions and Applications  
Week 5, Lec 1



So what will happen? So we fit the LDA parameters to document responses and you will get the topics and the coefficients topics, from the coefficients. So right and using these together given a new document you can estimate what is the response eta transpose times expected value of z bar given the words in the document, this we can estimate from your a SLDA model the same way you are doing from your LDA. So what the topic distributions is for a new document and this is what you get.

(Refer Slide Time: 31:49)



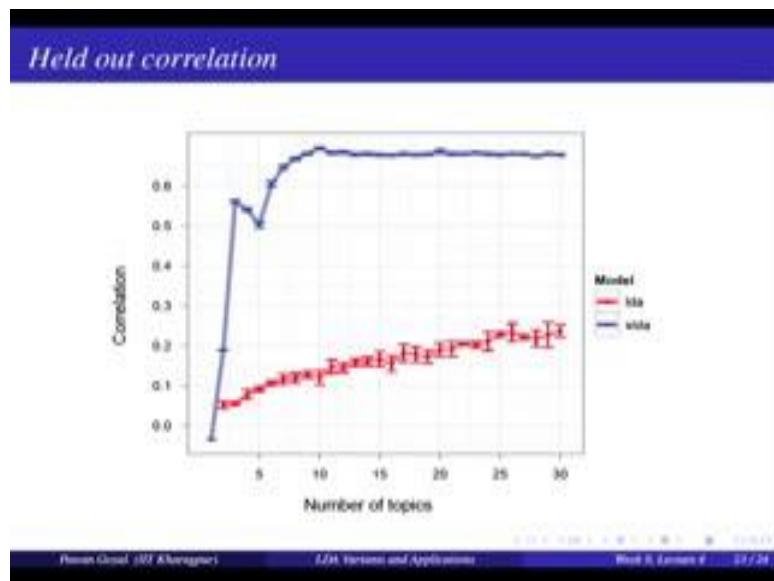
This is from the Pang and lee paper 2005.

So what there is they took 10 topic LDA model and put it on movie reviews. So you see here that 10 topics the most important words and they are plotted with their etas corresponding etas. So a high eta mean this topic corresponds to a high score or a high rating and in negative value of eta means this is called corresponding to a negative score negative rating. So what do you see here? On the higher side you have words like both motion simple perfect fascinating powering complex. So perfect in presenting a nice words that are coming with on the higher side and here you have least problem unfortunately supposed was flagged up and you will immediately. See they are like a negative image. So they are coming with a very low negative value, and this very nicely also puts your topics in a scale of minus 2 point plus that this was not available earlier.

Earlier you have different topics, but you did not know whether this topic is for a positive or negative rating.

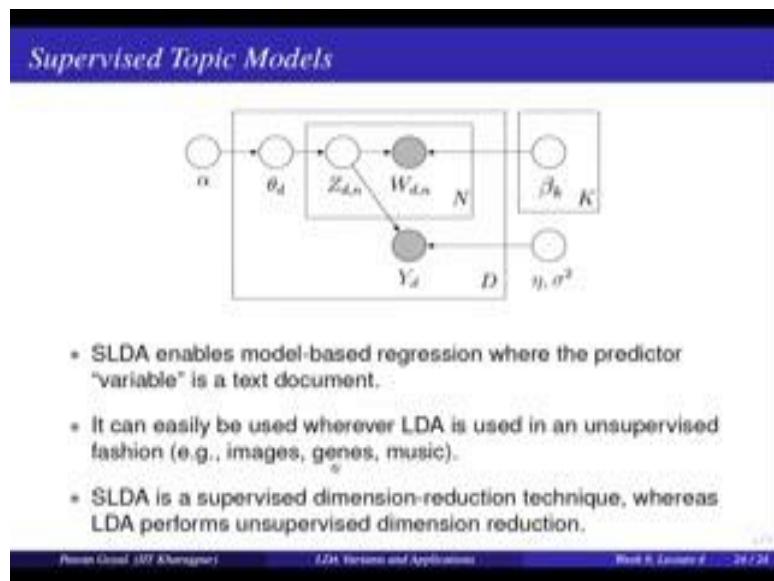
Now, you can also find out from your model itself.

(Refer Slide Time: 33:01)



And it gives a very good held out correlation also this is from number of topics; even if you increase the number of topics it gives you a much better correlation and the LDA model.

(Refer Slide Time: 33:12)



So what did you see here? So it enables model based regression where the predictor variable is a text document. So you did not have to run regression separately that is run inside the model; inside the model itself pure sampling your response by using a regression model.

Now, it can be used wherever LDA is used in an unsupervised fashion. So you can use it with images, music, etcetera wherever the data is paired with some sort of response variable. So you can also say that LDA is some sort of supervised dimension reduction technique, wherever it is a LDA is unsupervised technique right. You are seeing the response and by seeing that you are modeling your dimensional direction. So that is about using LDA. So there are some other variants also for topic models. So we will see some of those like the relation topic models and some simple intuition about the nonparametric basic models in the next lecture.

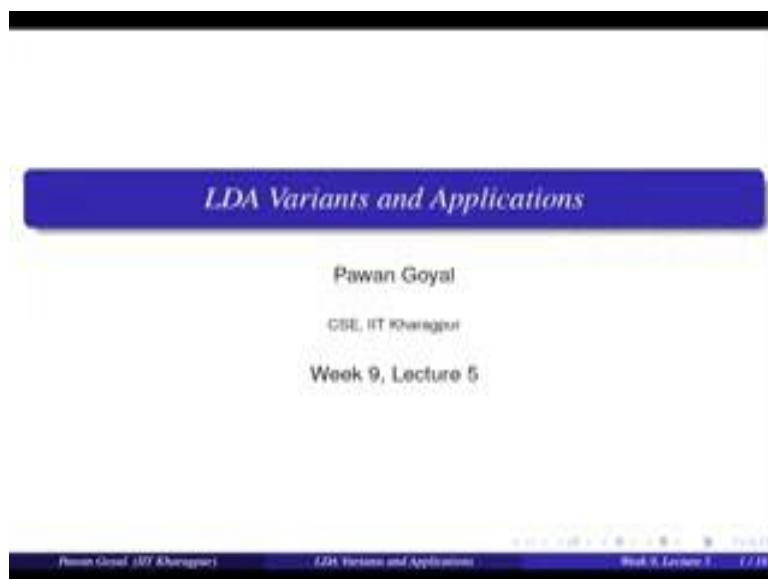
Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 46**  
**LDA Variants and Applications – II**

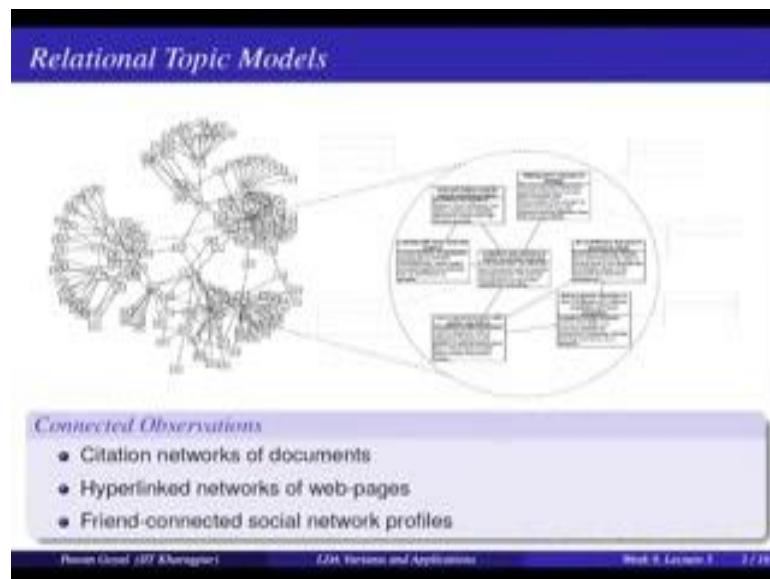
So, welcome back for the final lecture of this week. So, we are talking about different LDA variants and what are their applications. And we talked about three different variations like code rate topic models, dynamo topic models and sewage topic models. And we saw how they can model different assumptions we can also be used to pair the response as observation inside the model to make nicer topic models.

(Refer Slide Time: 00:44)



So, we now see in this lecture, we will also see some two other variants. So, one is called relational topic models, another is some sort of nonparametric Bayesian models, and you will see what sort of applications they can be used for.

(Refer Slide Time: 00:59)



So, what is the idea of relational topic models see? So, when you are talking about data, so many a times this is simple text data, so where they are not related to each other sometimes this is like a network also, so where different observations or different data points are connected together by some sort of graphic structure. So, for example, think about the scientific articles. So, as such you can think of them as separate different, different articles, this is one document, another document, different observations in fit a topic model there, or you can think of this as these are documents that are also connected.

What is the connection between the scientific documents with a citation network one paper might cite another paper. So, if it will be citing another paper, there is a link between them. So, you are seeing these observations are also connected. Similarly, think about the web page one web page might give a hyperlink to another web page. So, again these are observations that are connected. Then you can talk about friends then they are connected friends in the social network profile with the profile is kind of the document and the connection is like the so, and you are trying to model the connection.

(Refer Slide Time: 02:15)

*Relational Topic Models*

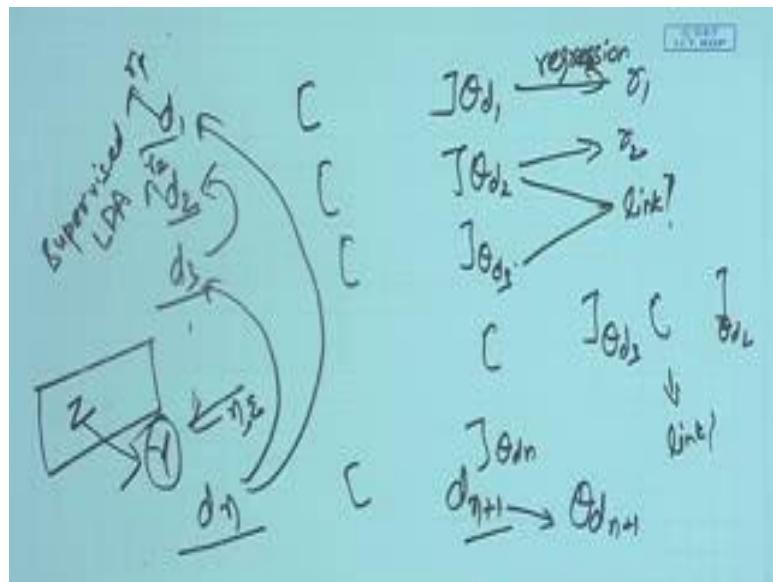
- LDA needs to be adapted to a model of content and connection
- RTMs find hidden structure in both types of data

Pawan Goyal (IIT Kharagpur) LDA Variants and Applications Week 9, Lecture 3 3 / 16

So, now what we are seeing here that can be adapt our LDA model where such that it can take care of not only the content, but also of this link that is also contained in connection both can this be handled by the topic models. And that is where you have this variation called a relational topic models or RTMs. So, relational topic models try to take care of this variation that I have the content as well as a connection.

So, before going to the model how we will try to do that. Suppose, we did not have this relational topic models what we are trying to predict here, I have this whole set of observations, now which two observations will be connected to each other. So, let us say I am talking about the scientific articles, when a person is writing a new article, what are the papers he will cite how will I model this. So, I will think of it as a learning problem, I will say ok, I have a dataset I know there are lot of papers and some papers writing another papers.

(Refer Slide Time: 03:27)



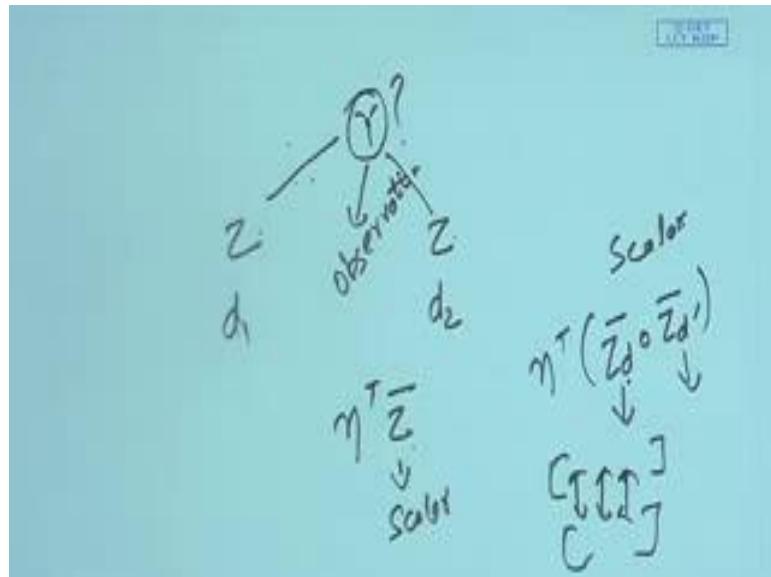
So, I will say ok, I have observations document  $d_1, d_2, d_3$  and so on  $d_n$ . I know  $d_3$  is citing  $d_2$ , I know  $d_n$  is citing  $d_3$ ,  $d_n$  is citing  $d_1$  and so on. I have these observations right. And I want to find out when a new able  $d_{n+1}$  comes in what will it cite, how will I solve this problem is suppose using LDA. What I will do? I will first in LDA over this corpus and I will find out this topic distributions  $\theta_{d1}, \theta_{d2}, \dots, \theta_{dn}$ . You find that out then I try to model given that this is  $\theta_{d2}, \theta_{d3}$ , will they link together or whether  $d_3$  will link to  $d_2$ . How will I model this?

Again you can use a regression here right. So, you can say given  $\theta_{d3}$  and  $\theta_{d2}$  predict the link, yes or no. So, I have some positive examples wherever there is link and negative examples where there is no link. At run time, when I get a new document using LDA, I get  $\theta_{dn+1}$ , now I try to combine it with  $d_1$  to  $d_n$  find out which one it is more likely to linked that is one way it can be handled. Now, does that remind your something, so that will remind if the now last topic that we discussed that was supervised LDA.

Remember there also we had the same problem, we said each document has a response. So, each document might have a response  $r_1, r_2$  and so on. So, we said ok one way could be take  $\theta_{d1}$  and run a regression from  $\theta_{d1}$  to  $r_1$ ,  $\theta_{d2}$  to  $r_2$  that was one possibility LDA plus regression. But we said we can do something smarter than that that is we take it as an observation within my model, and we say with each document with  $z$ , you are also

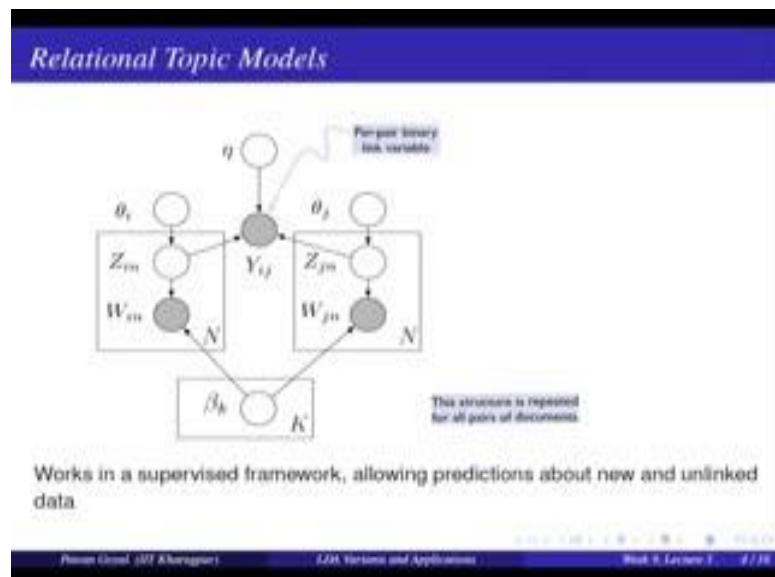
sampling your response variable with some eta and sigma. Now, can we use the same idea in this relational topic model also? So, the right now the difference is that we are working with the pairs.

(Refer Slide Time: 06:01)



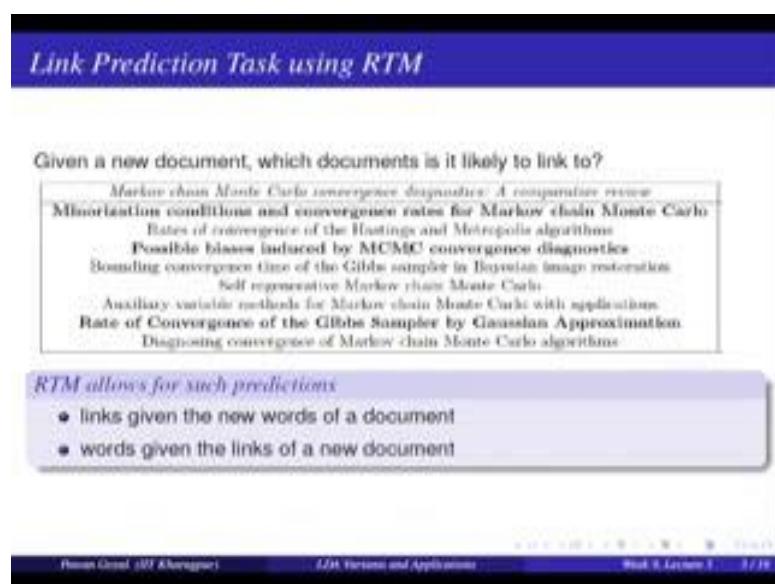
So, what we can do here? We can say that I have document d 1, I have document d 2. Now, this will have some z, this will have some z. Now, can I model whether they will link to each other? Given their individual topics and this can now be in observation inside my model. So, this is idea, this is a simple and direct extension of this supervised LDA that can I now use pairs and model the response as my another observation sorry model the connection as my another observation.

(Refer Slide Time: 06:41)



So, this is how it is done. So, you are having the same topic models sorry same model for so beta k the topic probabilities over different observations, but now you look at different pairs. So, let us look at these pairs document d i and d z. So, they have theta i, theta j variable. So, what your modeling given the z i n and z j n, their individual topic probabilities whether they will link to each other. So, there it is per pair this is a binary link variable; and this is indirect analogous to what we written in the case sewage topic models this eta times this into this plus some variance. So, this can also allow prediction about new and unlinked data.

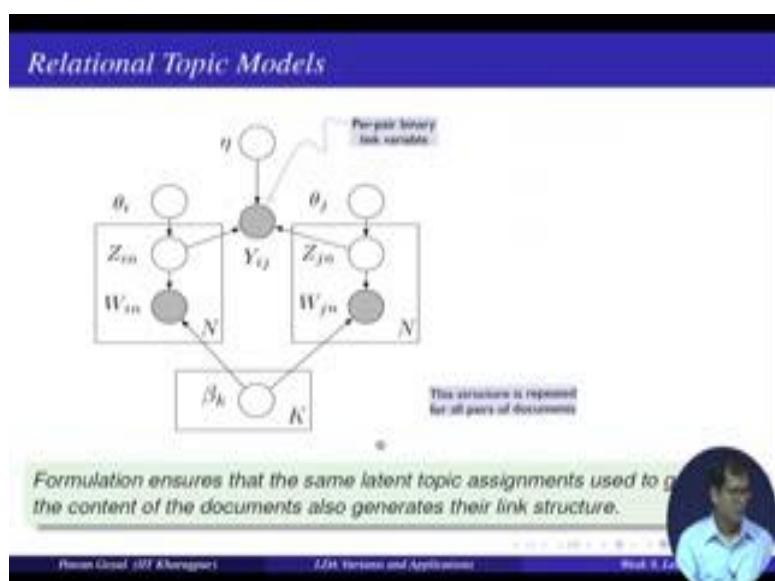
(Refer Slide Time: 07:31)



So, what are the examples, so how it will be used to something like this? Suppose, I have a top paper Markov chain, Monte Carlo convergence diagnostics a comparative review. So, and I want to find out what are the other papers it will link to or I can say suppose I am reading this paper, what are the other paper that are relevant to it like a recommendation problem. So, I can use this idea let what is the other papers that it will link to win my RTM model. So, here is some examples that this is what your RTM model gives and these are actually important papers that this document actually linked to.

So, RTM in general can allow the predictions that given a document with new words what are the document it will link to this is one thing that can be done. Also suppose you give a link that this document gives links to another document, what will be the approximate distribution of the document that also can be found using my RTM model. So, links given the new words of the document and words given the links of a new document both can be handled.

(Refer Slide Time: 08:34)



So, coming back to the model, this formulation what it ensures is that the same latent topic assignment is used to generate the content of the document and it also generate their linked structure. So, you are not separately learning the linked structure, you are learning it at the same time when you are learning your topic distributions, so that is interesting in this RTM model. So, from this  $Z_{in}$  and  $Z_{jn}$ , you are trying to predict what will be your  $Y_{ij}$ .

(Refer Slide Time: 09:10)

### RTM: Generative Model

1. For each document  $d$ :
  - (a) Draw topic proportions  $\theta_d | \alpha \sim \text{Dir}(\alpha)$ .
  - (b) For each word  $w_{d,n}$ :
    - i. Draw assignment  $z_{d,n} | \theta_d \sim \text{Mult}(\theta_d)$ .
    - ii. Draw word  $w_{d,n} | z_{d,n}, \beta_{1:K} \sim \text{Mult}(\beta_{z_{d,n}})$ .
2. For each pair of documents  $d, d'$ :
  - (a) Draw binary link indicator
$$y | z_d, z_{d'} \sim \psi(\cdot | z_d, z_{d'})$$

Pawan Ghosh (IIT Kharagpur) LDA: Variants and Applications Week 8, Lecture 3 3/78

Now, here is one problem. So, this is your simple model. So, this is what your LDA model is draw your topic proportions then for each word draw the topic assignment and the draw the word. Now, for the relational part, for each pair document d and d prime what you are doing you are trying to sample the variable y, whether they will link or not. And this depends on the z d and z d prime, and what is the function it is a psi of given z d z d prime.

(Refer Slide Time: 09:43)

### Link Probability Function ( $\psi$ )

Dependent on the topic assignments that generated their words,  $z_d$  and  $z_{d'}$ .

$$\psi_c(y=1) = \exp(\eta^T (\bar{z}_d \circ \bar{z}_{d'}) + v)$$

- $\bar{z}_d = \frac{1}{N_d} \sum_n z_{d,n}$
- $\circ$  notation denotes the Hadamard (element-wise) product
- Exponential function is being used, they also tried using sigmoid ( $\psi_0$ )
- Link function models each per-pair binary variable as a logistic regression parameterized by  $\eta_{1 \times K}$  and intercept  $v$  (in case of sigmoid)
- Covariates are constructed by the Hadamard product of  $\bar{z}_d$  and  $\bar{z}_{d'}$ , capturing similarity between the hidden topic representation of the documents

Pawan Ghosh (IIT Kharagpur) LDA: Variants and Applications Week 8, Lecture 3 3/78

And this is model like this. This is as an exponential function over eta transpose a Hadamard product over Z d and Z d prime. Now, what do I mean by that? So, remember in the sewage

LDA module also we did eta transpose Z bar, and this gave me a scalar - two vectors you are multiplying this giving you a scalar, you have the same dimensions. Right now here also you have eta transpose, same dimension has the number of topics and you are multiplying it with Z d bar Hadamard product with Z d prime bar. Now, what are Z d and Z d prime? Z d is what are the topic probability is doc document d and this is in document d prime again their vectors. Hadamard properties is nothing but element wise product, you multiply this element with this element this element with this element and so on and then you add it, and multiply with eta transpose. So, this will again give me a scalar and plus you might add some sort of bias or valence.

(Refer Slide Time: 10:56)

### Link Probability Function ( $\psi$ )

Dependent on the topic assignments that generated their words,  $z_d$  and  $z_{d'}$ .

$$\psi_c(y=1) = \exp(\eta^T (\bar{z}_d \circ \bar{z}_{d'}) + v)_+$$

- $z_d = \frac{1}{N_d} \sum_n z_{d,n}$
- $\circ$  notation denotes the Hadamard (element-wise) product
- Exponential function is being used, they also tried using sigmoid ( $\psi_0$ )
- Link function models each per-pair binary variable as a logistic regression parameterized by  $\eta_{1 \times K}$  and intercept  $v$  (in case of sigmoid)
- Covariates are constructed by the Hadamard product of  $\bar{z}_d$  and  $\bar{z}_{d'}$ , capturing similarity between the hidden topic representation of the two documents

Navigation icons: back, forward, search, etc.

Presentation footer: Pawan Ghosh (IIT Kharagpur), LDA: Variants and Applications, Week 8, Lecture 1, 8/19

This is a bias here. So, Z d is nothing but the topics of probabilities in each document use it for the pair and get a Hadamard product of those and plus with bias term gives you whether the document is likely to these two pairs are likely to link or not right, eta transpose and this total thing. And in the paper they also use the sigmoid function apart from the exponential function. And link function models each per pair binary variable as a logistic regression and this is by eta and nu.

(Refer Slide Time: 11:44)

Inference: How many links to model

- One can fix  $y_{d_1, d_2} = 1$  whenever a link is observed between  $d_1$  and  $d_2$  and set  $y_{d_1, d_2} = 0$  otherwise

Pavan Goyal (MIT Kharagpur) IITK Varian and Applications Week 9, Lecture 3 9/7/16

So, this is your relational topic model. What is one problem with this model? So, for each pair of documents you have to define their response that is whether they are linking together or not. So, why  $d_1, d_2$  is it 1 or 0, whenever document is giving a link to another document, you can always say to 1. But whenever document is not giving a link to another document, it is not necessary that they are not related at all, it may be that he is not giving the link or the link might occur in future there are many other things depending on the data points. So, one thing is that whenever there is a link you take it as 1, whenever there is no link you take it as an unobserved variables. So, I do not know if there is a link or not, so that also helps you in being computationally efficient, it avoids getting you so many different pairs of document.

(Refer Slide Time: 12:41)

**Inference: How many links to model**

- One can fix  $y_{d_1, d_2} = 1$  whenever a link is observed between  $d_1$  and  $d_2$  and set  $y_{d_1, d_2} = 0$  otherwise
- Problem with that approach is that the absence of a link cannot be construed as evidence for  $y_{d_1, d_2} = 0$
- So, in these cases, these links are treated as unobserved variables
- Also provides a significant computational advantage

*In large social networks like Facebook, the absence of a link between two people doesn't necessarily mean that they are not friends.*

Pawan Goyal (IIT Kharagpur) JDA, Variations and Applications Week 9, Lecture 3 37/38

So, that is especially in the case of social networks, whenever there is an absence of link you cannot take it as if  $y_{d_1, d_2}$  is equal to 0. Example also you can see from your like Facebook profile. So, you are taking the profile as your documents, and you are finding out if this person will become a friend of another person. So, even if they are not friends at this time it might happen that they will become friends in future. So, absence of the link cannot be taken for that their response should be 0. So, it can be taken as in an unobserved variable that is a better solution.

(Refer Slide Time: 13:18)

**Predicting links from documents**

<i>Markov chain Monte Carlo convergence diagnostics: A comparative review</i>	(4) PDF
<b>Minimization conditions and convergence rates for Markov chain Monte Carlo</b>	
Rate of convergence of the Hastings and Metropolis algorithms	
Possible biases induced by MCMC convergence diagnostics	
Bounding convergence time of the Gibbs sampler in Bayesian image restoration	
Self-regenerative Markov chain Monte Carlo	
Auxiliary variable methods for Markov chain Monte Carlo with applications	
<b>Rate of Convergence of the Gibbs Sampler by Gaussian Approximation</b>	
Diagnosing convergence of Markov chain Monte Carlo algorithms	
Exact Bound for the Convergence of Metropolis Chains	
Self-regenerative Markov chain Monte Carlo	
<b>Minimization conditions and convergence rates for Markov chain Monte Carlo</b>	(4) PDF
Gibbs-markov models	
Auxiliary variable methods for Markov chain Monte Carlo with applications	
Markov Chain Monte Carlo Model Determination for Hierarchical and Graphical Models	
Melding instrumental variables	
A qualitative framework for probabilistic inference	
Adaptation for Self-Regenerative MCMC	

Given a new document, which documents is it likely to link to?

Pawan Goyal (IIT Kharagpur) JDA, Variations and Applications Week 9, Lecture 3 38/38

So, this is done and then so they tested whether this words better than the model of LDA plus regression. This was the person that we saw that you take the topics from document d 1 document d 2 and learn a regression over there that is one option. Another is you take the pair which are linking as an observed variable and also use it for learning your topics inside the model itself. And that they found gives much better performance than LDA plus regression model for this task given any document what is the document it will linked. So, this is a other examples like for this document competitive environments evolved better solutions for complex task documents like coevolving high level representations were coming out to be at first position in the RTM model, but were not coming out from LDA plus regression model. So, this was something interesting that there saw from this model.

(Refer Slide Time: 14:16)

The slide has a blue header bar with the title "Nonparametric Models". The main content area contains a bulleted list:

- In LDA, we need to specify the number of latent clusters (topics) in advance.
- In many data analysis settings, however, we do not know the number and would like to learn it from the data.
- Bayesian Non-Parametric (BNP) models assume that there is an infinite number of latent clusters, but only a finite number of them is used to generate the observed data.

At the bottom of the slide, there is a navigation bar with icons for back, forward, search, and other presentation controls. The text "Powered by OpenOffice.org" and "LDA: Versions and Applications" are visible on the left, and "Wolfram R, Kenney E... 32 / 39" is on the right.

Now, finally, we will wrap up this week by some small some simple discussion on non-parametric Bayesian models. So, what do we see in the case of LDA models? So, in LDA, what we are having we are defining that this corpus consists of certain topics; and then each document I find out what are the topic assignment is what I find out what is the topic assignment for that word. So, what is one problem here, I have to tell a priori how many topics that are there in the data, but that may not be an easy parameter to always give right, you may not know should I use 10 topics, 20 topics, 100 topics in this data.

So, can my model itself find out what will be the number of topics as such and that is where we come to this topic of non-parametric Bayesian. So, this parameter goes away you do not

have this parameter of number of topics and anymore. So, in many data analysis settings, we will not know what is the number of topics a priori. So, in Bayesian non-parametric models, what we assume is that there is an so we are not fixing number of parameters, but that also means my data can have any number of parameters that means, any number of topics. So, I have to start with the assumption that there are infinite number of topics, and my data can take any finite number of them. So, it can take 100, 500, 30 whatever it means.

So, in generally I have an infinite number of topics, so that is what is interesting about it assumes that there are there is an infinite number of latent clusters or topics, but only a finite number of them is used to generate the observed data. But you do not give it as an input to your model how many such questions we need this is what model on its own comes up when it sees the observations.

So, the posterior provides the distribution over the number of clusters, the assignment of data to clusters and the parameters of each cluster all these are different, different parameters of your LDA model that can be learned from the Bayesian non-parametric models. So, we will see so there are actually many different variations of this Bayesian non-parametric models also known as hierarchical Dirichlet processes because this is a very wide topic we will not cover this topic in detail in this course. We will just give you some intuition that how do you come up with this infinite topics in your setting, and how does it model any finite number of topics in the data even if you are starting with infinite topics. So, we will talk about one particular setting that is a Chinese restaurant process that can be used for modeling or grouping your observation into some finite number of groups the number is not told a priori. So, what is this process, it is quite interesting.

(Refer Slide Time: 17:21)

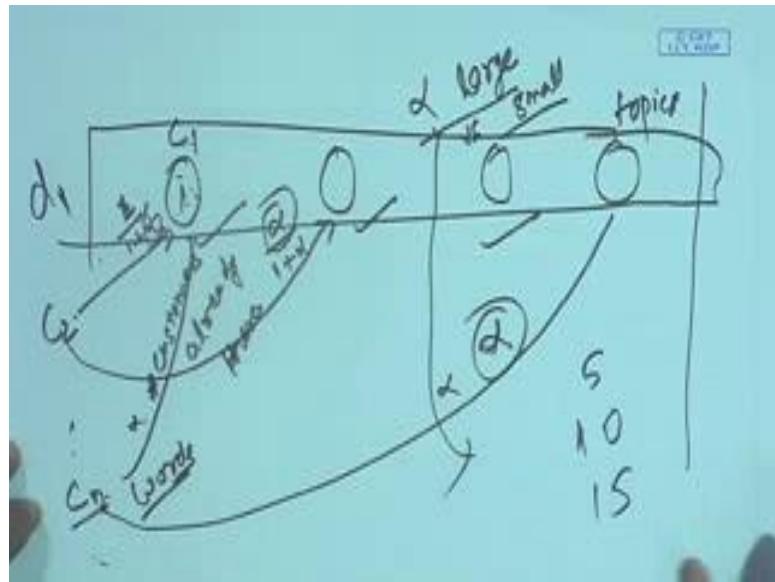
**Chinese Restaurant Process: the prior over groupings**

- Imagine a restaurant with an infinite number of tables
- Imagine a sequence of customers entering the restaurant and sitting down
- The first customer enters and sits at the first table.
- The second customer enters and sits at the first table with probability  $\frac{1}{1+\alpha}$  and the second table with probability  $\frac{\alpha}{1+\alpha}$ .
- When the  $n$ th customer arrives, she sits at any of occupied tables with probability proportional to the number of previous customers sitting there, and at the next unoccupied table with probability proportional to  $\alpha$ .
- A large value of  $\alpha$  will produce more occupied tables (and fewer customers per table).

So, the Chinese restaurant process, so we are talking about the restaurant. So, we have saying that suppose this is a restaurant where there are infinite number of tables when you see the here the term infinite immediately what comes to your mind, these are the latent clusters. So, I have infinite tables in that restaurant. And then there are customers that are coming up. So, we are like observations. So, we are fitting the observation to top to different clusters.

Now, the model says when the customers are coming in the restaurant, how are the seated on the tables. Now, one thing that it says is that the tables are of infinite capacity. So, they can hand each table can take as many customer as you want. So, there is no limit on the number of customers that can sit on a table. So, now how do we assign customers to the tables in this process and that is where so it says ok.

(Refer Slide Time: 18:20)



So, I have some tables infinite number of tables. So, customer one is comes in and sits at the first table, table 1 fine. So, customer 1 is no problem. Now, the customer 2 comes in. So, idea is that customers 2 now sits on table 1 with the probability 1 divided by 1 plus alpha, and sits on a new table with a probability alpha divided by 1 plus alpha here c is add adds up to 1. So, it can sit at either table 1 or table 1. Now, when a new customer comes in certain points of C n, it can sit of suppose this table, this table, this table is occupied; it can sit either at the occupied tables or add a new table. So, what they say? It can sit on any of the occupied tables with a probability proportional to number of customers already there, and on a new table, with the probability proportional to alpha.

So, remember that is what we did here this is proportional to 1; this is proportional to alpha, and then we normalize we get one divided by 1 plus alpha, alpha divided by 1 plus alpha. At any given a point of time, you will have certain configuration different tables will have different number of customers. So, new customer comes in, it can sit of any other table with a probability proportional to the number of customers already there or it can choose a new table and that is how you can fill in any number of customers on n number of tables. So, you will see at the end of this process, you can have any number of tables. So, random process you might have 5 tables filled, 10 tables filled, 15 tables filled. So, you can have any number of clusters, this does not have to be defined a priori. From your sampling, you can choose any number of topics or clusters

Now, one thing what will be the effect of the parameter alpha. Suppose, your alpha is large versus alpha is a small. If your alpha is large, what will happen? If alpha is large when a customer comes in, there is a large probability that will choose a new table so that means, with large alpha you can get large number of topics. So, is it gives you some idea. If you want more number of topics and each topic having small number of observations, then you will choose a large alpha, but if you choose a smaller alpha, you will probably choose a small number of topics because if the customer will have a higher probability of sitting at any of the tables that are previously occupied. So, that will be the effect of these alpha.

So, coming back, so first customer enters and sits at the first table; second customer enters and sits at the first table with the probability of 1 by 1 plus alpha and the or the next unoccupied table with the probability alpha divided by 1 plus alpha. And when the another customer arrives, she sits at any of the occupied tables with a probability proportional to the number of previous customers already seated on the table, and at the next unoccupied table with the probability proportional to alpha, and that is how I can distribute all the customers in a restaurant. And we saw that if I have a large value of alpha, I will have more occupied tables and fewer customers per table. So, we saw that.

(Refer Slide Time: 22:17)

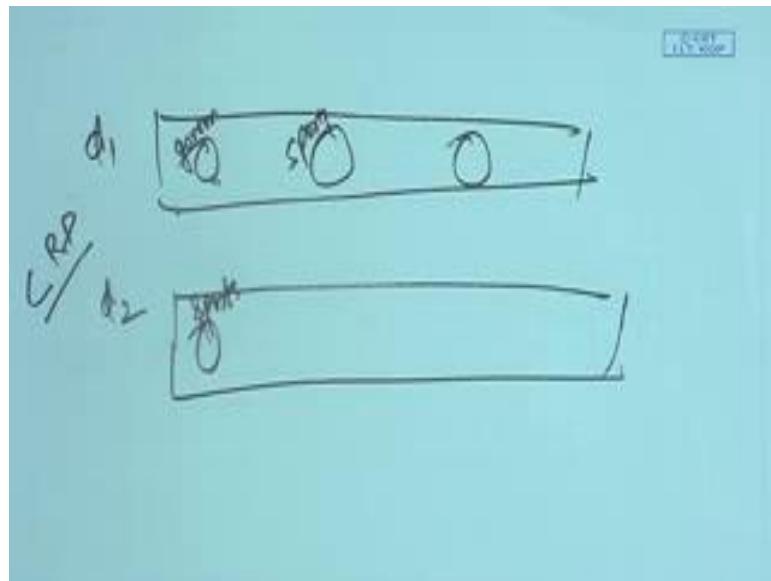
*How to model a corpus*

- CRP helps in obtaining a random partition as the sequence of customers sitting at tables in a restaurant.
- Tables can be thought of as 'topics' and customers as 'word' in the restaurant (document)
- CRP, however, does not model the entire corpus.
- For that, we extend CRP to a set of restaurants, each for one document,
- This model is known as Chinese Restaurant Franchise (CRF)

Now, it looks interesting, if I have a document I can model. So, these tables as my topics and these customers as my words and I can say which words are assigned to what topics. But how do I model my whole corpus you see I can model my one this can be thought of as my one

document d 1 I know what are the topics, and what are the customers. But suppose I have a corpus right that, so generally I wanted for my corpus. So, corpus means I will have multiple such documents. Now, one thing would be I fill it independently for each document, but if I do that there is no correlation between the assignment here in d 1 and d 2.

(Refer Slide Time: 23:10)



So, it might happen d 1, I have first table some words are filling in here like government, table two some words like a sports something. And document d 2 the word is sports might start coming here and there will be no correlation between topics here in this document, topics in this document. So, I cannot run this Chinese restaurant process independently for different documents that way I will not be able to have a correlation between these. So, what is actually done?

So, we saw that Chinese restaurant process it helps in obtaining a random partition as the sequence of customers sitting at tables in a restaurant. So, we get a random partition of words into various topics. Now, we also sit tables can be thought of as topics and customers as word in the restaurant or restaurant can be thought of as document. However, it cannot model the entire corpus for that we can extend CRP to a set of restaurants, so that is can be extended to defining each document as a separate restaurant and this is called Chinese restaurant franchise. So, now, we are going from Chinese restaurant process to Chinese restaurant franchise. So, this franchise has a lot of restaurant and that is why we will try to connect the different topics. So, let us see how do we connect the different topics.

(Refer Slide Time: 24:39)

**Chinese Restaurant Franchise (CRF)**

- Customer in the  $j$ th restaurant sits at tables in the same manner as in CRP, and this is done independently in the restaurants.
- But then, how do we achieve coupling among restaurants?
- The coupling is achieved by a franchise-wide menu.
- The first customer to sit at a table in a restaurant chooses a dish from the menu and all subsequent customers who sit at that table inherit that dish.
- Dishes are chosen with probability proportional to the number of tables (franchise-wide) which have previously served that dish.

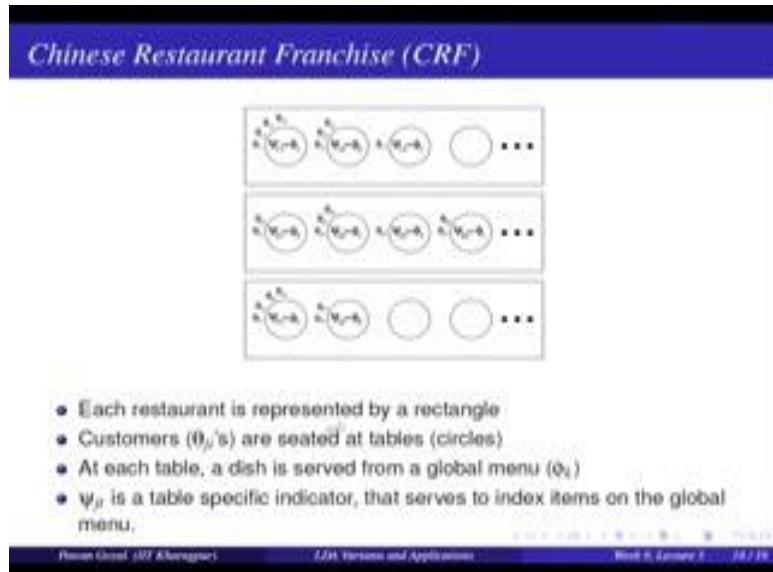
So, again so this is what is similar to CRP. So, customer in the  $j$ th restaurant sits at tables in the same manner as in CRP. So, customer comes in the restaurant, any of the restaurant now it sees which tables already occupied how many customers are there, the probability proportional to the number of customers is sits at that table, and the probability proportional to alpha it sits at the next unoccupied table, so that remains the same as CRP.

Now, where is this connection among topics coming and that is coming in from the nice idea about the franchise that is let us say that they just franchise wide menu. So, now, you have a new term of menu, there is a certain dishes that are there in all the all the restaurants and that is franchise wide. So, whenever a customer comes at a given table, it orders a dish from that menu, now whoever comes in next, we will have the same dish whoever customer comes in next we will share the same dish. Now, this dish is shared throughout the restaurants. So, now, this dish at a table acts as the topic of that document. So, now, the dishes are same across the across the different restaurants and that is why you can model it the same topics.

So, how do we achieve coupling among the restaurants. So, this is achieved by a franchise wide menu. So, what we say that the first customer to sit at a table in a restaurant chooses a dish from the menu, and all the customers was sitting at that table will share the same menu. And dishes are chosen with a probability proportional to the number of tables which already have that served that dish. So, if this dishes serve at many different tables across the franchise

then this will have a higher probability, so that way you are also giving you prior probability on choosing a particular dish on a given table.

(Refer Slide Time: 26:51)



So, it will look something like that. So, what you are seeing? You are seeing three different restaurants in this franchise 1, 2 and 3. Now, this psi 1 1 tells for the restaurant one what is the dish at table 1. So, this is phi 1 this is from the global menu of dishes phi 1, phi 2 and so on it is a global menu. The second table has phi 2, third table has phi 1; that means, more than one tables can order the same dish that is possible in that model. So, why because you see, when a customer comes in it will first choose the table. So, it can either to choose one of the table that is occupied on your a right table, then when you choose the table it chooses one of the dishes so that means, it can the same dish can be chosen at more than one tables. So that means, this table also the same dish was chosen and so on.

Next, second restaurant, first table choose dish phi 3, second table phi 1, third table phi 3 and so on. So, it has certain different are choices and then the third restaurant first table choose phi 1 and second table choose phi 2 and so on. Now, you also seeing the customer assignments. So, in this first restaurant customers are coming in theta 1 1, theta 1 2, theta 1 3 and so on; theta 1 1 sits a table 1; theta 1 2 sits a table 2, theta 1 3 at table 1, 1 4 table 2, 1 8 at table 1, 5, 6 here, 7 here and so on. So, there are certain assignments that are given to these customers.

Similarly, here theta 2 1, 2 2 sit here and 2 3, 2 4, 2 6 sit here. Now, similar to LDA what we will have when a customer chooses a dish, it will see what the other customers have chosen that is where they will be coupling that the words will be assigned similar sort of topics across the different restaurants. So, you are seeing each restaurant is modeled by a rectangle, each customer are seated at the tables, and each table you are serving a dish from a global menu. So, this is a table indicator. So, by modeling this, now you will know that what are the assignments of your different words to different topics in all these documents; interesting thing is that you do not have to tell how many topics do you need a priori, what is the model can itself learn from the observation.

So, what we have seen? We have seen ok, you have an LDA model you can use it for very nice applications like finding out similarity between documents, words, but suppose you want to use some rich assumption like topics are correlated changing over time. So, you can modify that model that is why it is it has a lot of then the model is very, very strong, it can use many different variations. Then you say ok, I this is unsupervised model, but suppose I want to model some responses with that that is what is the rating of this text or how many likes I can take it as an observation variable inside response variable and learn my topics accordingly that it is sewage topic model. I can go further and say which two documents are connected together that I can again measure the link as an observation, so link can be as a response or observation link two pair of documents. This was by relational topic models.

And then we said ok, suppose we do not want to use any parameters how many how many topics etcetera then you go to Bayesian non-parametric. And again there a lot of different models we talked about in very briefly about Chinese restaurant process and Chinese restaurant franchise, but they are the other variations also that can be used. And depending on your application, you can choose one of the other models and they are tools available that will allow you to model any of this. So, that brings us to the end of this ninth week where we discussed a lot about topic models, and also about semantics, token semantics.

From the next thing, we will start talking about different applications. So, we will talk about entity linking information extraction in the next week, and there we will go about other applications in the subsequent weeks.

Thank you and see you in the next week.

**Natural Language Processing**  
Prof. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

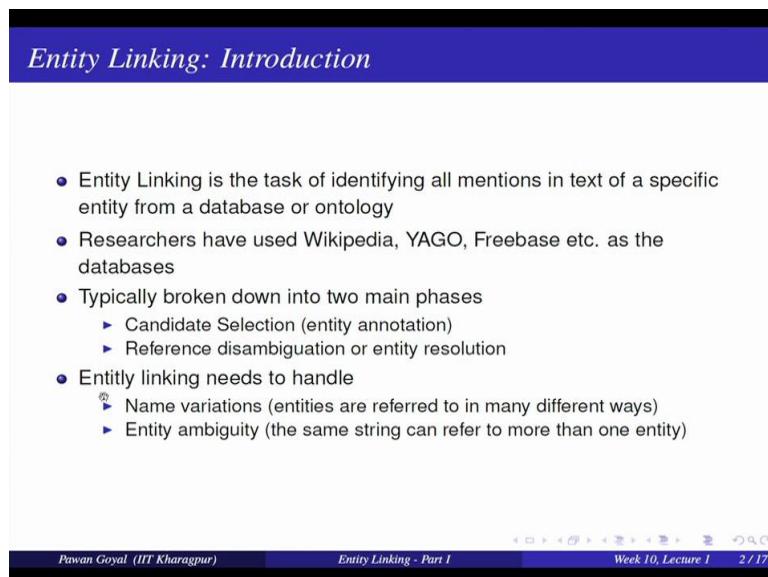
**Lecture - 47**  
**Entity Linking – I**

Hello everyone. Welcome back to the week 10 of this course. So, we have been doing a lot of basics in this course and last week we finished our discussions on semantics. And during semantics we also talked about various applications where you can use topic models and other things like distribution semantics and (Refer Time: 00:35) semantics.

So, starting from this week we will focus mainly on applications. So, you will use both the basics that we have covered and some new methods for solving very specific tasks. And for this course we have chosen the tasks that are very very popular in the field of NLP and text panel. So, this week we will start with; so we will start the first two lectures which is entity linking and then we will go towards information extraction.

So, today we are starting with entity linking.

(Refer Slide Time: 01:06)



*Entity Linking: Introduction*

- Entity Linking is the task of identifying all mentions in text of a specific entity from a database or ontology
- Researchers have used Wikipedia, YAGO, Freebase etc. as the databases
- Typically broken down into two main phases
  - ▶ Candidate Selection (entity annotation)
  - ▶ Reference disambiguation or entity resolution
- Entity linking needs to handle
  - ▷ Name variations (entities are referred to in many different ways)
  - ▷ Entity ambiguity (the same string can refer to more than one entity)

Pawan Goyal (IIT Kharagpur) Entity Linking - Part I Week 10, Lecture 1 2 / 17

So, what is entity linking as such? What is the problem there? So we were discussing sometimes in the starting and in some lectures that when we encounter text data lot of entities are mentioned there. And it would be helpful for a task if you what are the different entities

that are used in this particular text data. Now there are various knowledge basis and resources where you have a good list of all these entities, and some descriptions of these available.

So, suppose you can find out from a text what are the important entities here and if you can link them to a knowledge base, then you can do a lot of further inferences over these. For example, this entity is a person and there in the database you will have a lot of information about this and you can make use of all these information. So, this as if you are having some background knowledge about various entities given a text data when you encounter the entity you try to link it to the knowledge base and then you can extract the background knowledge about it. And use that for different task in this particular data.

So, this is a problem of entity linking. So, we can define it in this manner that entity linking is the task of identifying all mentions in text of a specific entity from a database or in ontology. So, what we are assuming here we have a database or we can also call it knowledge base ontology, where I know what are the different entities that I need. And with that information with that entity I will have some different sort of information. For example in Wikipedia, think of all the Wikipedia pages you have, you can call, you can think of all these Wikipedia pages at the entity pages and you have lot of information about the entities in each page.

So, this is my knowledge base. Now when you encounter a text there your problem is find out if a particular n gram or a sequence of phrases sequence of words together correspond to an entity in the Wikipedia. And if so then link it to that Wikipedia page and this is the overall idea of entity linking problem.

So, lot of different databases can be used. For example researchers have used Wikipedia lot then YAGO, freebase etcetera. And the task of entity linking when you are doing you can break it into two different phases. So, one phase would be you find out from the text what are the appropriate candidates or entities that should be linked. So, this is called the entity mention detection part. Identify what are the mentions of entities that should be linked to the database. And once you found out what are the important entities the next one would be to appropriately link it to the database that is the second part; reference disambiguation or entity resolution.

Now why would that be a problem? See same as we discussed in the case of words is disambiguation. With the same entity name there might be different reference. So, for example New York, it can be New York City and there might be movie with the name New

York. There might be tv serial with the name New York. So, when in a text you have a mention of New York you want to know whether you want to link it to the New York City or the film or tv series or something else. So, there is a problem of disambiguation here. And this is also to be handled when we are solving the problem of entity linking. Find out what are the appropriate entities, and then appropriately link them to their entries in the database.

So, what are the things your challenge is that one needs to handle in this problem? So, one is name variations. So, the same entity can be written in many different ways. For example, simple things like Hillary Clinton. So, in a text you can all you might write it with only the name Clinton with the last name Clinton or maybe there is a middle name also involved here. So, you can write it in different different manners. So, your problem is you have to handle all these name variations and all these should map to the appropriate entity in the knowledge base.

And then the second challenge is entity ambiguity; that is the same string can refer to more than one entity. So this is also we discussed, New York can refer to multiple entities. So, both these challenges are to be handled in the problem of entity linking

(Refer Slide Time: 05:37)

*Entity Linking: Introduction*

- We will take Wikipedia as the knowledge base to understand the task.
- With Wikipedia as the knowledge base, this task is commonly known as *Wikification*.

Pawan Goyal (IIT Kharagpur) Entity Linking - Part I Week 10, Lecture 1 3 / 17

So now, for this course what we will do we will take one particular database that is Wikipedia as our base database. So, we will always link a text to Wikipedia. So, this will be our database by default for this lecture in the next lecture. But in general you can use any other database and you can accordingly modify your approaches for that. So one particular

terminology; so if you are using Wikipedia as your knowledge base then this task of entity linking is called Wikification or Wikifying. So, you are taking a text and you are trying to Wikify that. That means, find out the entities that are important and then linked them to their appropriate Wikipedia pages; that is the problem of Wikification.

So, now let us going detail about this problem, what is the different processing involved and what are the different techniques you can used.

(Refer Slide Time: 06:31)

The screenshot shows a web application titled "What is Entity Linking?". At the top, there's a language switch from Italiano to English. Below it is an "Input Text" area containing a physics abstract. The abstract discusses degeneracy removal due to a geometric gradient onto a metasurface, mentioning spin optics, helicity, and dispersion engineering. On the right side of the input text, there's a vertical bar labeled "Many links" at the top and "Few links" at the bottom, with a slider between them. Below the slider are "Reset" and "TAGME!" buttons. Below the input text is a "Tagged text" section where the abstract is displayed again, but with certain words underlined and colored blue, indicating they have been linked to Wikipedia pages. The bottom of the slide shows navigation icons and the text "Pawan Goyal (IIT Kharagpur)", "Entity Linking - Part I", "Week 10, Lecture 1", and "4 / 17".

So, here is one example of entity linking or Wikification as such. So, on the top what you are seen, you are seeing a research article from physics domain. And you are having; so this like an abstract here, so you are having text such as degeneracy is removed due to a geometric gradient onto a meta surface and so on. So, there is a text involved here. Now what is the task you are trying to Wikify that? So that means, you are trying to identify what are the important entities and what are their pages in Wikipedia.

So, if you see on the bottom you are finding various links. So, spin optics. So, where optics is linked to some different page; an example is from degeneracy. So, this is linked to degenerate energy levels that are a page in Wikipedia. and you can also find out some specific content about that page, if you just take your mouse over there. And that is being done to many different words here; optics, control, light, photon, helicity, angular momentum and so on.

That means, these words are the appropriate mentions and they are then linked to their Wikipedia pages. So, this is the process of Wikification.

(Refer Slide Time: 07:52)

The screenshot shows a news article from a website. The title is "Iranian POW negotiator holds talks with Iraqi ministers". Below the title is a paragraph of text. In the middle of the text, the word "Baghdad" is highlighted with a blue box, indicating it is a hyperlink to a Wikipedia page. At the bottom of the slide, there is a navigation bar with icons for back, forward, and search, and text indicating the speaker is Pawan Goyal, the lecture is "Entity Linking - Part 1", and it is Week 10, Lecture 1, slide 5/17.

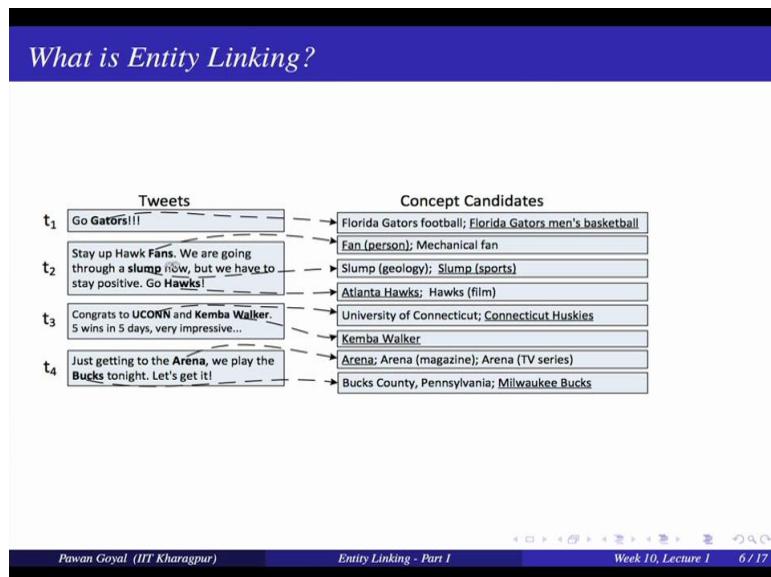
Similarly, here you are seeing a news article. In the article with the news you are seeing various words are hyperlinks. So, there you can click this word and go to their appropriate Wikipedia page. Also for example, here in Baghdad; so it will open the Wikipedia description of the word Baghdad and you might just want to read the summary or if you want to know more you can click on this opening Wikipedia and go to the Wikipedia page.

So, so we can also see why this might be very important application. So, you can talk about reading news articles or scientific articles also any other text it can be tweet text. So, when you reading the article there might be many different terms that are very very specific domain and if you are not in expert in the domain you will not know what these terms stand for.

So, then what you would do you will take this terms and try to search in the dictionary or some on Google or some or may be on Wikipedia, and that will require a lot of time from your side to fully understand that article. So, what is Wikification doing? It is helping you in that given a text it will automatically identify what are the important entities and it will link the Wikipedia pages. So, it will avoid, it will do all the tasks for you and you can even see the description in the same page or you can go to the Wikipedia page and read more about it.

So, that helps a lot in enhancing your reading for a certain news article or scientific article. Additionally it can also help if you are planning two certain tasks on that text. Your own to get some semantics from the text then also having this knowledge that this entity corresponds to this Wikipedia entity might help you in getting some knowledge from the Wikipedia page, and take it as a feature for your task. So, this is also one other application for this Wikification.

(Refer Slide Time: 09:55)



So, we have seen for scientific articles and news articles you can also do it for very short text like tweets. So, tweets have very little context. So, here you are seeing with four different tweets “Go Gators” and here the problem is what does this ‘Gator’ refer to. So, in Wikipedia there are two different mentions that are possible Florida Gators football of Florida Gators men’s basketball. In this particular context it refers to Florida Gators men’s basketball. And you want to link it to that particular entity.

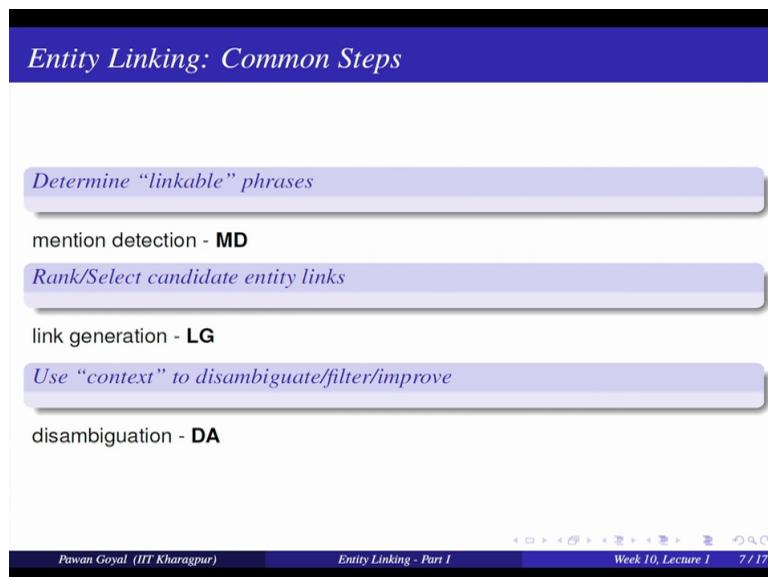
Similarly here; “stay up Hawk Fans. We are going through a slump now, but we have to stay positive. Go Hawks”. So, here you have entities like fans, slump and Hawks and they need to be linked to appropriate entities. Again you are seeing their multiple possibilities and you need to find out what is the appropriate mention Wikipedia; same with the other examples that you are seeing here.

So, you can do it for also for short text like tweets. With tweets there is a very little information and you want to find out what is the appropriate entity that this tweet links to.

And you can also think of many other applications on your own where this entity linking is important.

So, now how is actually done? So for that let us try to understand what are the different phrases again, what do we need to do systematically.

(Refer Slide Time: 11:13)



So, what are the common steps? First step is it will determine what are the linkable phrases, and this step is also called Mention Detection. That is from the text find out what phrases what n grams are to be linked. For example, we were seen words like Baghdad and here Gators what the mentions that word to be linked to the knowledge base. And this approach for detecting these words is called mention detection.

Now once we have found out what are the appropriate mentions for to be linked then what will be the next step. Next step would be you have to identify what are the possible candidates to which it can be linked. Like in the previous slide we were seen Gators can link to two different entries. So, identify what are the possible entries, and this is called the link generation part. So, you have to select what are the candidate entity links and what are the all the links you have to list somehow this is called link generation part.

Now, once you know what are all the links then what will be the next step; you have to find out what is the most appropriate link for all these set. And this will call the disambiguation part. So, this is use the context to disambiguate what is the appropriate link it this entity

should be link to and you might also want to filter you might want to improve your whole task. And we will see some examples for all these; how do you filter and improve your task based on this.

So, these are three main steps: sometimes you might combine the first two steps also, that is when you are detecting the mentions you also finding the candidates at the same time. So, that is also possible.

(Refer Slide Time: 13:03)

Mention Detection (MD)

... degeneracy is removed ...

Pawan Goyal (IIT Kharagpur) Entity Linking - Part I Week 10, Lecture 1 8 / 17

So, let us see these steps again in the context of Wikification. So, you are having a text where you have this sentence degeneracy is removed. And there are some words before and after. So, what is mention detection? Find out that the word degeneracy is to be linked is the appropriate mention, so that is in green in this slide. So, that is a mention detection part. Then second part would be link generation.

(Refer Slide Time: 13:25)

The screenshot shows a search interface with a query "Q ... degeneracy ...". Below the query, four thumbnail images of Wikipedia articles are displayed, each titled "Degeneracy". The first three thumbnails correspond to "Degeneracy (mathematics)", "Degeneracy (biology)", and "Degeneracy (graph theory)". The fourth thumbnail, which is highlighted with a green border, corresponds to "Degenerate energy levels". At the bottom of the slide, there is a footer bar with the text "Pawan Goyal (IIT Kharagpur)", "Entity Linking - Part I", "Week 10, Lecture 1", and "10 / 17".

Find out all the appropriate, all the possible phrases pages in my database. So, here you can see the degeneracy occurs in mathematics, in biology, in graph theory and degenerate energy levels. So, there are four possible links. So, then you have a task of disambiguation. That among the four links what should be the appropriate page to which this entity should be linked, and that will be the third step and that is the disambiguation. And you will say this is the fourth one degenerate energy levels is the appropriate entity page for my mention of degeneracy. And these are three steps for this entity linking

(Refer Slide Time: 14:09)

The screenshot shows a Wikipedia article titled "United States". The left sidebar contains a list of basic elements: "Basic element: article (proper)", "But also" (with sub-points: redirect pages, disambiguation pages, category/template pages, admin pages), and "Hyperlinks" (with sub-points: use "unique identifiers" (URLs), [[United States]] or [[United States|American]], [[United States (TV series)]] or [[United States (TV series)|TV show]]). To the right of the sidebar, the main content area displays the "United States" article, including its summary, sections like "History", "Government", "Economy", and "Demographics", and a "See also" section. At the bottom of the slide, there is a footer bar with the text "Pawan Goyal (IIT Kharagpur)", "Entity Linking - Part I", "Week 10, Lecture 1", and "11 / 17".

Now, so we might like to understand; what are the some of the basic features of Wikipedia that can help us in designing an algorithm for Wikification. So, Wikipedia all of you know about Wikipedia and you have been reading Wikipedia for many of your for knowing different concepts and all. So, what do you seen Wikipedia there is a page like that there is a title and certain texts about the page and you see there are various links also. So, there is an article then additionally they can be some redirect pages. So, you might have come across you are searching for something in Wikipedia and it redirects you to some other page in Wikipedia. So, these are also lot of redirect pages in Wikipedia.

Then there are disambiguation pages. We will see an example in the next slide then there are category template pages that allocate. What are the different categories in Wikipedia this category, what are the subcategories and then there are some admin pages. Now what is important for our task is that there are lots of hyperlinks in Wikipedia. So, what hyperlinks in Wikipedia? So, different words and phrases are linked within the Wikipedia itself. So, we will see that in Wikipedia article certain concepts have a hyperlink and you click on the hyperlink and you will go to the corresponding Wikipedia page. So, there are lots of in links and out links that are going on within Wikipedia.

So, United States for example; whenever you have a phrase like United States you may have a link saying it is linking to the United States TV serial or American TV show etcetera. So, you will find if you will see the source this will be like that of the hyperlink. So, you will have a double parenthesis to denote what is the appropriate entity inside the source. And that if you see the source wise you can find out. And these are various hyperlinks that you have in Wikipedia.

(Refer Slide Time: 16:23)

The screenshot shows a Wikipedia disambiguation page for 'New York'. The title bar says 'Preliminaries: Disambiguation Pages'. The main content area is titled 'New York (disambiguation)'. It states 'New York is a city in the United States of America.' and 'New York may also refer to:'. Below this, there are several sections with categories and sub-links:

- Places**:
  - New York, New York
  - New York, North Dakota
  - New York, Mississippi
  - New York, Minnesota
  - New York, Missouri
  - New York, Nebraska
  - New York, Nevada
  - New York, North Carolina
  - New York, Oregon
  - New York, Pennsylvania
  - New York, Texas
  - New York, Virginia
  - New York, Wisconsin
  - New York, Wyoming
  - New York County, New York
  - New York City, a city in New York State and the largest city in the United States
  - New York City, a former county surrounding New York City and its six boroughs
  - New York County, covering the same area as the New York City boroughs of Manhattan, Bronx, Brooklyn, and Queens
  - Province of New York, a British colony comprising the state of New York
  - West New York, New Jersey, a town across the Hudson River from New York City
  - West New York, New Jersey, a city in Bergen County, New Jersey
  - New York, Missouri, a German community founded in 1850 or 1856 near Columbia, Missouri
- Media and entertainment**:
  - Black Sea
  - Film
    - New York (film), 2009 Hollywood film directed by Kader Khan
    - New York (film), 1997 film directed by Martin Scorsese
    - New York, A Documentary Film, by Ric Burns
- Literature**:
  - New York (poetry collection), 1966 poetry collection by Philip Larkin

At the bottom, it says 'Pawan Goyal (IIT Kharagpur)', 'Entity Linking - Part I', 'Week 10, Lecture 1', and '12 / 17'.

Then you have a lot of disambiguation pages. For some entities where a lot of different mentions are available you might have also categories in disambiguation; that in the category of entertainment what are the possibilities, in the category of politics what are the possibilities and so on. So, like here you have for the entity New York disambiguation page, so you will have places what are the possibilities, then media entertain entertainment. And you will see that in each category there are lots of entities that correspond to this the main entity New York.

So, you will see in these cases is the same single entity can map to may be 15-20 different pages in Wikipedia, and they are nicely categorized in this disambiguation page. Categories may or may not be there in various pages. So, what do we need to about the whole architecture? Lot of pages in Wikipedia and each page has some name that will be the identifier; you may be the identifier then lot of text involved in the text you have various hyperlinks, where different phrases are linked to their own Wikipedia pages.

And some entities will have their own disambiguation page, where you will find out what are the different ways in which this entity can be used, this maybe also under various categories.

(Refer Slide Time: 17:36)

*Some Statistics*

**WordNet**

- 80k entity definitions
- 142k senses (entity - surface forms)

**Wikipedia**

- 4M entity definitions
- 24M senses

Pawan Goyal (IIT Kharagpur) Entity Linking - Part I Week 10, Lecture 1 13 / 17

So, once you know about that let us say some small statistics. So, that is we talked about WordNet for sense disambiguation. We given a sentence, we want to find out each word; what is the appropriate sense in WordNet it corresponds to. So, in WordNet how many entities we had? We had roughly 80k; 80000 different entity definitions, and 142000 different senses.

On the other hand Wikipedia is much larger repository. So, in Wikipedia overall there are 4 Million entity definitions and this keeps on increasing, and there are 24 Million different senses. So, is much much larger in comparison to WordNet. So, our task is from all these 24 Million senses find out given a text what are the entities import that are important and which of the sense they correspond to.

(Refer Slide Time: 18:34)

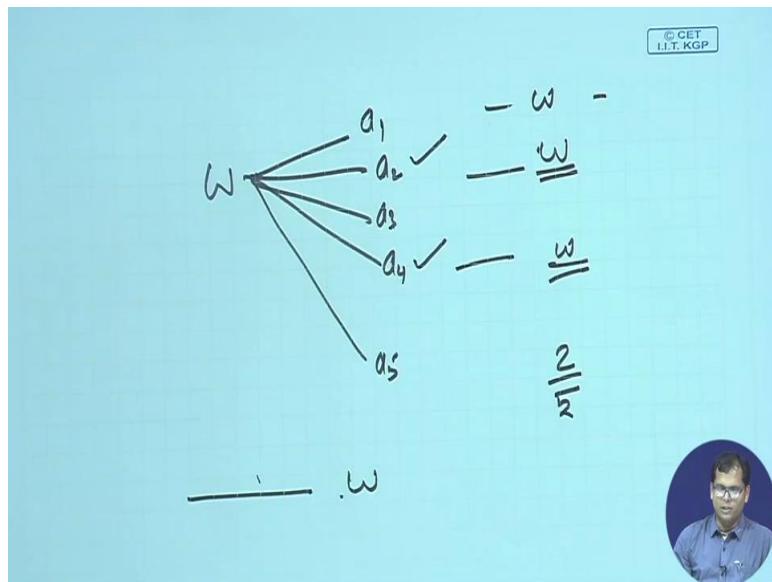
The slide has a blue header bar with the text "Wikipedia based methods". Below it is a pink rounded rectangle containing the question "What can be a good measure for MD?". A light purple rounded rectangle contains the text "keyphraseness( $w$ )" and its definition: "Number of Wikipedia articles that use it as an anchor, divided by the number of articles that mention it at all.". To the right of this text is a mathematical formula:
$$\frac{CF(w_l)}{CF(w)} \longrightarrow \begin{array}{l} \text{Collection frequency} \\ \text{term } w \text{ as a link to another} \\ \text{Wikipedia article} \end{array}$$
A downward arrow points from the fraction to the word "Collection frequency" and "term  $w$ ". In the bottom right corner of the slide, there is a circular profile picture of a man with glasses and a blue shirt. At the bottom of the slide, there is a navigation bar with icons and text: "Pawan Goyal (IIT Kharagpur)", "Entity Linking - Part I", and "Week 10, Lec 1".

Now let us see; what are the simple measures that we can think for the three steps or let us say only the two steps: the mention detection and disambiguation. Mention detection, that is in a text whether a given n gram is an appropriate mention or not. So, what we will do initially we will see some sort of measures that can be taken simply by the Wikipedia structure or Wikipedia data. So, let us see.

So, let us talk about this mention detection part, whether a particular phrase is a good candidate for a mention. So, what will be a good measure for this? So, if you think about using Wikipedia structures we can say that- ok Wikipedia has lot of pages. Suppose I find out this particular n gram how many times it occurs in Wikipedia; and among whatever times it occurs, what fraction of times it is actually linked to something. So, what is the idea if a word is linked much more number of times; that means, it might be a good candidate for mention. If it is not linked many times; that means, this may not be a very good candidate.

And this very simple criteria that you can use from Wikipedia; so in this is called the keyphraseness of a word or also a phrase. So, number of Wikipedia articles that use it as an anchor divided by the number of articles that mention it at all.

(Refer Slide Time: 20:12)



That means, I will take a word w and I will find out what are all the Wikipedia pages where it occurs. So, it occurs suppose in five (Refer Time: 20:19) articles: article 1, article 2, article 3, 4 and 5. And among the five articles say 4 and 2 provide a link with this w; so where w occurs with the link to some Wikipedia page. And a 4 also this w occurs with the link to a Wikipedia page, but a 1, a 3, a 5 do not provide a link. So, here w occurs without a link.

So, what is this keyphraseness? Keyphraseness is what fraction of the page in Wikipedia is it linked wherever it occurs. So, five linked to six keyphraseness will be 2 by 6. And that is a good measure in that it will tell me whenever encounter a new phrase w how many times it is actually linked in Wikipedia and use that to detect if it is a good mention at all. So, this is a very simple measure. So, we will say how many times it occurs and among whatever time it occurs how many times it is linked to another Wikipedia article.

Now here we do not worry about whether it is linked always to the same Wikipedia article or multiple Wikipedia articles, the only thing is it is linked to something then we will considerate in the numerator. So, this is the keyphraseness for a word.

(Refer Slide Time: 21:41)

*Wikipedia based methods*

What can be a good measure for DA?

*commonness(w, c)*  
The fraction of times, a particular sense is used as a destination in Wikipedia.

$$\frac{|L_{w,c}|}{\sum_{c'} |L_{w,c'}|}$$

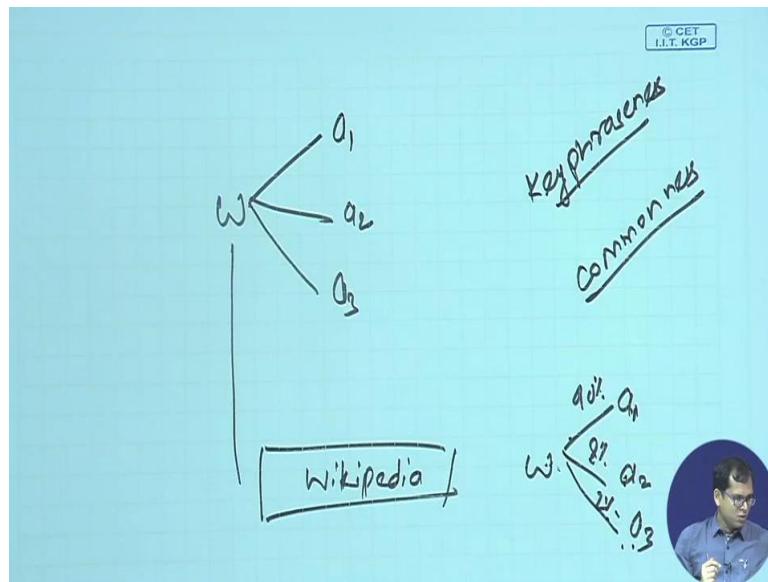
Number of links  
with target  $c'$  and anchor text  $w$

Pawan Goyal (IIT Kharagpur) Entity Linking - Part I Week 10, Lecture 1 15 / 17

$$|L_{w,c}| / \sum_{c1} |L_{w,c1}|$$

Now, what can be a good measure for disambiguation? So now, let us again think about it can we use Wikipedia to find out a good measure for disambiguation. So, what can be the simplest measure that you can think of? So, I have a word and it can correspond to multiple entities.

(Refer Slide Time: 22:10)

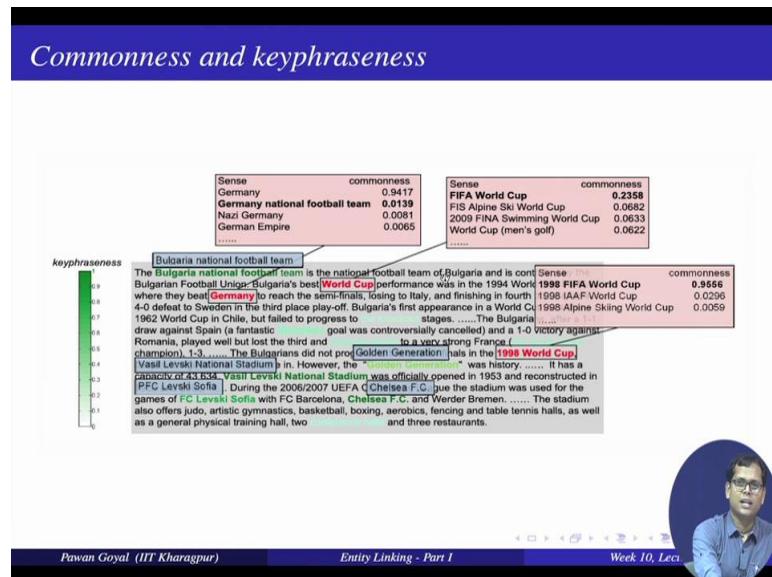


So for example; let us say I have a word w and in general it can link to three Wikipedia pages: a 1, a 2, a 3 all are possible reference for this Wikipedia page. Now, what can be a good baseline to find out what is an appropriate disambiguation page? So, for that I can again use Wikipedia. So, I will see in the whole Wikipedia whenever w is link to something, so link to these a 1, a 2, a 3 what fraction of times it is link to a 1 suppose it link to a 1 90 percent of the time, a 2 80 percent of the time and a 3, 2 percent of the time.

And this can be a good measure to say 90 percent of times w links to the article a 1. So, by default I will say w will link to a 1. So, that can be one simple measure and this is called the Commonness; so this is commonness. So, I will define the commonness for a word and a concept; concepts here are three concepts, three Wikipedia concepts. And what is the definition? So, the fraction of times a particular sense is used as a destination in Wikipedia. So, number of times word is link to c divided by number of times word is link to any c prime

So like here it will be 90 divided by 100. Suppose they are 90, 80 and 2 pages, so 90 divided by 100 is the commonness for w and a 1 80 by 100 is commonness for w a 2 and 2 by 100 is commonness for w and a 3. So, this is another simple measure. So, you have you have seen keyphraseness and commonness. And they are simple measures it is direct from Wikipedia.

(Refer Slide Time: 24:06)



So, now let us see one example. So, here is one text that is like a report of a match, and what you are seeing? You are seeing words they are coloured and colours depend on the keyphraseness score; that is from 0 to 1. So, dark green is keyphraseness 1; that means, it is

always linked in Wikipedia. So, here like Bulgaria National Football team is roughly always linked to Wikipedia it has a high keyphraseness. Some words like here the knock out are not always linked they have a very low keyphraseness.

Now what about the commonness? So now you will take a particular entity like here Germany. And you will see; what are the all candidates like Germany, Germany National Football team, Nazi Germany, German Empire. Similarly for world cup it can be FIFA World Cup, FIS Alpine Skiing World Cup- 2009, FIN Swimming World Cup, World Cup Men's Golf etcetera; these are a various can candidates. And you are computing commonness by seeing how many times this word is actually linked to these entities divided by the number of times it is actually linked and this gives you the commonness.

So, Germany is linked to the Germany the word Germany like 95 percent of a time Germany National Football team 1.39 percent of the time and so on; similarly for FIFA World Cup. So now, from there you can choose by default the word the sense with the highest commonness. So, like 1998 FIFA World Cup will come up here, FIFA World Cup will come up here; but they will a problem with this entity. This Germany will written the word Germany 95 percent time, but in this case the appropriate mention is Germany National Football team and it will not be able to detect this.

So, this is the idea about keyphraseness and commonness. And clearly you can see if there is a 1 there is one problem with this approach. So, is it always the best decision to use either the only the commonness for linking their particular entity. So what do you say from whatever we saw in the last page? Is it always the best decision to use commonness? It cannot be right, because what you are seeing whenever a word w occurs in any context I will always assign it to a 1 by default because it has the highest commonness.

That means, I have never using the context in which the word w occurs, by default I am assigning it to the category or link a 1. So that means, I will always make some mistakes right, there will be some pages at least weight should link to a 2 or a 3 and in those cases I will (Refer Time: 27:03) link it to a 1 by default. So, I cannot design a very good system by this approach. There is always the chance of making mistakes because you are taking the default case. And this also corresponds to the one of the baseline that you can use in words sense disambiguation. That is you take sense of a word that is most probable sense. That is

like a baseline, but this will never help you in designing a very good system, because you are doing it independent of the context you are always linking it to this page.

So, now what we will see in the next lecture is that can we also use the context to improve this method. Instead of using commonness can we use something from the context to find out; among the three what should be the appropriate link for this particular entity.

(Refer Slide Time: 27:57)

The slide has a dark blue header bar with the text "Always the best decision?" in white. The main content area is white. At the bottom, there is a dark footer bar with the text "Pawan Goyal (IIT Kharagpur)", "Entity Linking - Part I", "Week 10, Lecture 1", and "17 / 17".

- This can never help you build an accurate system, because you will always give some wrong links.
- Need to use the context.

So, what did we see? So commonness and keyphraseness are simple measures, they can help you to design a good baseline that will work most of the times, but cannot help you build an accurate system, because you will always give some wrong links. And we can see why, because there is a default to the most probable link. And whenever the word is used in not so probable links it can never be correctly assigned.

So, we need to use the context and that is what we will see in the next lecture; that how do we use the context from the word to disambiguate the links.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 48**  
**Entity Linking – II**

Welcome back for the second lecture of this week. So, we have started talking about entity linking and we talked about two different approach; so one approach where we use the keyphraseness in commonness to find out what are the appropriate mentions and how do you link to them their corresponding reference in the knowledge base. And we were considering Wikipedia as our knowledge base. And we found out one particular problem with using simple keyphraseness and commonness.

(Refer Slide Time: 00:50)

The slide has a blue header bar with the title "Keyphraseness and Commonness: Always the best decision?". Below the header, there are two main sections:

- Depth-first search**: A box containing a snippet from Wikipedia about Depth-First Search (DFS). It explains that DFS is an algorithm for traversing or searching a tree or graph. It starts at the root and explores as far as possible along each branch before backtracking. Formally, DFS is an uninformed search that progresses by expanding the first child node of the search tree that appears and thus going deeper and deeper until a goal node is found, or until it hits a node that has no children. Then the search backtracks, returning to the most recent node it hadn't finished exploring. In a non-recursive implementation, all freshly expanded nodes are added to a LIFO stack for exploration.
- Sense-based approach table**: A table comparing the commonness and relatedness of various senses for the word "tree". The columns are "sense", "commonness", and "relatedness". The data is as follows:

sense	commonness	relatedness
Tree	92.82%	15.97%
Tree (graph theory)	2.94%	59.91%
Tree (data structure)	2.57%	63.26%
Tree (set theory)	0.15%	34.04%
Phylogenetic tree	0.07%	20.33%
Christmas tree	0.07%	0.0%
Binary tree	0.04%	62.43%
Family tree	0.04%	16.31%
...		

Below the table, a pink box contains the heading "Using Relatedness: Basic Idea" and a bulleted list:

- In a sufficiently long text, one finds terms that do not require disambiguation at all.
- Use every unambiguous link in the document as context to disambiguate ambiguous ones.

At the bottom of the slide, there is footer text: "Pawan Goyal (IIT Kharagpur)", "Entity Linking - Part II", "Week 10, Lecture 2", and "2 / 9".

So, what is the problem? So, in commonness what we were doing, we were always taking the page that is having the highest commonness. So, what will happen? Suppose I have word like tree, and the commonness to the sense tree is 92.82 percent, but the other concepts because they occur rarely commonness is like 2.94 percent, 2.57 percent and so on. So, what you are seeing wherever the word tree occurs you will by default assign it to the first sense of tree, and you will not look at the context at all.

So, in this case, so you have it article about depth first search and you have a sentence where the word tree occurs, and because of using commonness, you will always assign it to the

sense tree. But the correct sense says here is tree data structure, but it has a very small commonness, so only 2.57 percent. So, what you need to do for assigning it to the correct data structure correct sense there is a tree data structure. For that you should be able to use the context here that is what are the referent words that I am seeing in the context.

Now, question is how best we can use the context. Now, I do not want to use some random words in the context, I want to use the words in the context that have actual correspondence to a Wikipedia page, so that I can find out something about the Wikipedia page how common this Wikipedia page is to one of its references. Now, there we fall into the same problem that how do I use a Wikipedia reference to any of its pages when the disambiguation has not yet happened, so I am only at the stage of disambiguation. So, how do I use the actual entities, entity page?

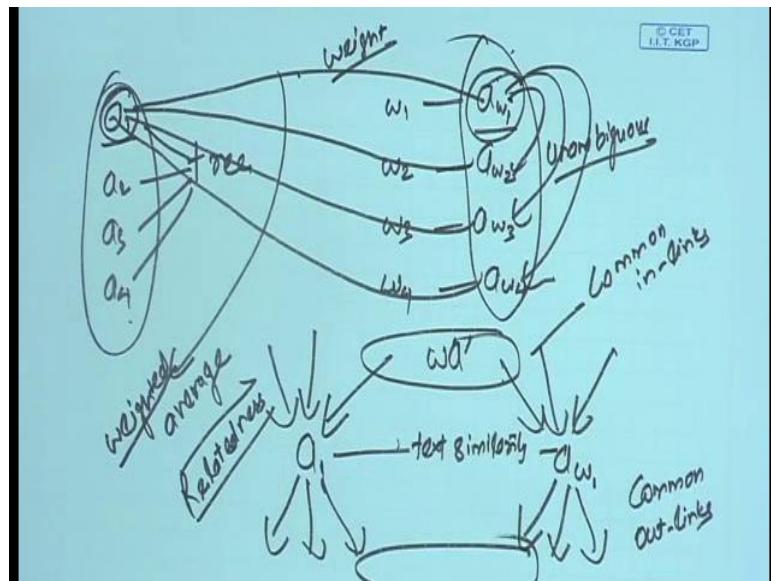
And for that, a nice tree can be used. And the idea is ok, so with this word there are many other words that are coming in this article or this piece of text, and some of these will be appropriate mentions, and they will point to one or many Wikipedia pages. Now, some of these at least some will be there that have a unique disambiguation page in Wikipedia. So, there is a unique page in Wikipedia where they link to. And there I do not need to do any disambiguation. So, why do not I use only those pages which have a unique page in Wikipedia to find out what should be a good of a Wikipedia page for this entry tree, so that is what is done. So, this is the hypothesis that if you have a sufficiently long text, you can find out terms that do not require disambiguation at all; there will be some terms that have only one mention or one referent in Wikipedia.

Now, use this unique unambiguous link in the document that context to disambiguate the ambiguous ones. So, what is the idea here, so you are given this article and you want to find out what is the appropriate sense for this word is. So, in a sense, I mean what is the appropriate link in Wikipedia, is it tree, tree graph theory, tree data structures, set theory etcetera. So, what I will do I will see what are the other mentions in this page. So, what are the other things you have seen algorithm, tree structure, graph, backtracking, uninformed search, tree backtracks, LIFO stack and so on. Now, among these there will be some that have only one Wikipedia page as the referent, so these are shown as box search.

So, algorithm, tree structure, uninformed search and LIFO stack have only one Wikipedia page. So, I will take these unambiguous links and try to find out how close these four links

are to my all of these possible senses. So, how close are these four links to this possible sense and the one sense that is having the closest to these four will be called by (Refer Time: 05:11) sense, this will be the link to will to which I will link my corresponding mention.

(Refer Slide Time: 05:19)



So, how do we compute the relatedness score, and this can be very simple. So, you can initially start by representing each candidate sense and context term by a single Wikipedia article. So, for example, what is happening now, so a word like tree, and tree corresponds to many different senses and call them your article 1, article 2 article 3, article 4. And if you remember your word sense disambiguation this, it is like constructing various sense backs, there are four different senses they are like sense back. Now, you are having a context back kind of context track, where you are saying ok in this context of tree I am finding four different words word 1, word 2, word 3, word 4. Now, what is the property of each word, they link to one page in Wikipedia, so like a w 1, a w 2, a w 3, a w 4. So, now, these are Wikipedia articles and they are also Wikipedia articles.

(Refer Slide Time: 06:43)

The slide has a blue header bar with the title "Computing Relatedness". The main content area contains a bulleted list of four items:

- Each candidate sense and context term is represented by a single Wikipedia article.
- Thus the problem is reduced to selecting the sense article that has most in common with all of the context articles.
- Comparison of articles is facilitated by the Wikipedia Link-based measure, which measures the semantic similarity of two Wikipedia pages by comparing their incoming and outgoing links.
- The relatedness of a candidate sense is the weighted average of its relatedness to each context article.

Below the list is a pink rectangular box containing the text "How to give different weights to the context terms?". At the bottom of the slide, there is a navigation bar with icons for back, forward, and search, along with the text "Ptwan Goyal (IIT Kharagpur)", "Entity Linking - Part II", "Week 10, Lecture 2", and "3 / 9".

Now, the problem is select the sense article that is the most in common with all of the context articles, so that is among the four articles, which is most common with all these four articles. And you will see what is the argmax that is having the more similarity with these four articles. And this can be captured in many different ways. So, we will talk about one particular method. So, one particular method is you just take the Wikipedia link based method that is two pages are similar if they are having many incoming and outgoing links common.

So, what is the idea so how do you find out how similar a 1 is to a w 1. So, I will say I have two articles a 1, a w 1 in Wikipedia, I find out what are the incoming links to this article, and what are the outgoing links from here; same I will do for this article. And now once I have found this out, I can see ok, what are the common links. So, how many articles in Wikipedia article w a prime are linking to both of these. Similarly, what are the articles to which both of these links to? And these are very good measures for finding out how similar they are. You see we can always do it by seeing how similar they are by measuring their text similarity. How much text similarities is there, you can capture cosine similarity and what or something else.

But this is a nice link based measure that says ok, how many pages linked to both of these, so what are the common in links, and how many pages they both linked to that is a common out links. And this is again a nice measure in that it says ok; this article refers to both of these

that means they need to have something common similar they both mentioned the same article again that means, they need to have something in common. And you will find out how many what fraction of incoming links are common, what fraction of outgoing links are common and that you will take it as a measure for computing how similar these two articles are. And this is also called by relatedness. So, we talked about keyphraseness, commonness and this is relatedness.

And then you can find out the relatedness of a candidate size sense by taking a weighted average of its relatedness with all of the context articles. So, that is to find out the relatedness of this sense a 1, you will say ok, it is relatedness with a w 1 with a w 2, a w 3, a w 4 by using this measure, then you take a average or a weighted average - computed weighted average. And this will be what is the relatedness of this sense a 1, a 2, a 3, a 4 whichever as the highest, you take that. Like, if you see the previous slide, so here you were capturing showing relatedness of various senses. And this tree data structure had the highest relatedness 63.26 percent that uses the average of relatedness with all these the four different context articles.

Now, so there is one term here we are taking a weighted average. Now, what should this weight depend on, why should I weight one of this higher than the other ones. So, again if you think about it, it can depend on which context term is more important than another. So, what we have done we have taken the context, we have found out to all the words that are mentions; and from there whichever are so these were unambiguous, whichever were an unambiguous we are taking them as my context to find out the relatedness. But some of these might be more important for this topic other documents than others. So, can I give a weight depending on how important they are to the topic?

Now, the question again comes how do I know which one are more important to the topic or the theme of this document. And there you see you can again use the idea of relatedness that means one among these four context senses or articles the one that is having the highest relatedness with the others which is more appropriate to the theme of the document. Yes, because there is a theme of the document and the words that are appropriate should also be connected to each other that means, a word that is having a high relatedness with other words is it should be given a high values, and that is a nice method to also give a weight here. Weight to different of these relatedness that is how related these this context article is to the other context articles.

(Refer Slide Time: 12:22)

**Weighting the Context Terms**

- **link probability:** Use the ones that are almost always used as a link within the articles where they are found, and always link to the same destination
- **relatedness:** We can determine how closely a term relates to the central document by calculating its average semantic relatedness to all other context terms

*These two variables - link probability and relatedness - are averaged to provide a weight for each context.*

Pawan Goyal (IIT Kharagpur) Entity Linking - Part II Week 10, Lecture 2 4 / 9

So, in general, there are so what are the things that are used to give a weight to the context term, so one is called link probability. So, what is link probability, so again in your context, you are finding say four or five articles where the link is unambiguous, there is only one link. Now, you can use the link probability itself that is among the four which one is like a keyword that is it always links to something. Some words may not link may not always link some words always link. So, the words it always link should be given a high weightage, because I know this is a more a specific term if a word is sometimes links sometimes not linked it may not be a very important keyword, so that can be one measure. What is the link probability, probability that it will be given a link in general Wikipedia that is same as my keyphraseness measure? And certain thing we have already discussed that is relatedness.

So find out how closely it relates to the central document by computing it is average relatedness to all other context terms. So, I take for each word what is the link probability or keyphraseness and relatedness. Now, I have two different measures, so how do I take these together to compute the weight of this word, you can simply take an average to provide the weight for each context, so is that clear now. So, you have some words in the context they are unambiguous. For each word, for each context word you find out the relatedness of this mention in your mention sense one of the sense, taken weighted average and this weight depends on what is the link probability, and what is the relatedness with all the context terms. And by doing this method you can find out relatedness of each of the four senses.

Now, so we have discussed, we can take some mentions by some method, and then once with the mentions we can also give a link by finding out which of the candidates are similar in from with the context mentions. But here there is an interesting question that by using all this when we are doing all this approach, can I go back and also improve my mention detection part, so how I was detecting the mentions.

(Refer Slide Time: 15:03)

*Can we improve mention detection with this approach?*

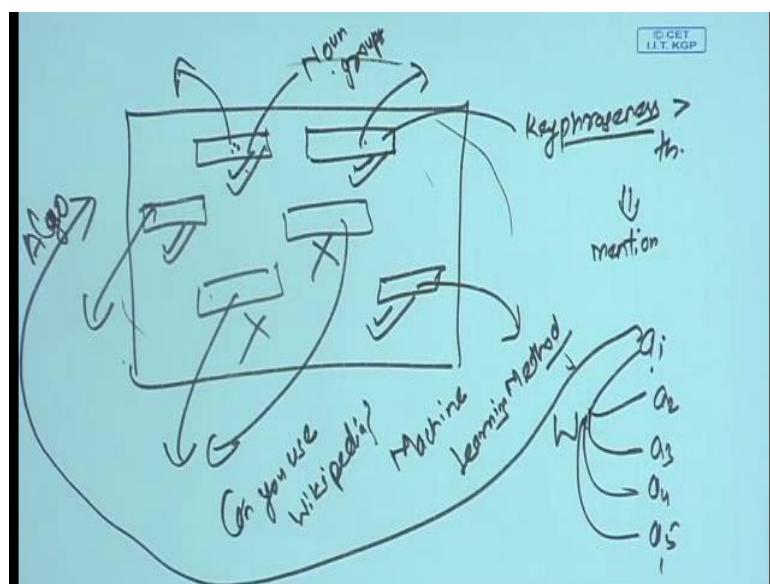
- The link detection process starts by gathering all n-grams in the document, and retaining those whose probability exceeds a very low threshold. *Is it the best method?*
- All the remaining phrases are disambiguated using the approach mentioned earlier.
- This results in a set of associations between terms in the document and the Wikipedia articles that describe them.

*Can you use this to learn – which concepts should be linked?*

Pawan Goyal (IIT Kharagpur) Entity Linking - Part II Week 10, Lecture 2 5 / 9

So, I was gathering all the n-grams in the document, and retaining only those whose probability exceeds a very low threshold.

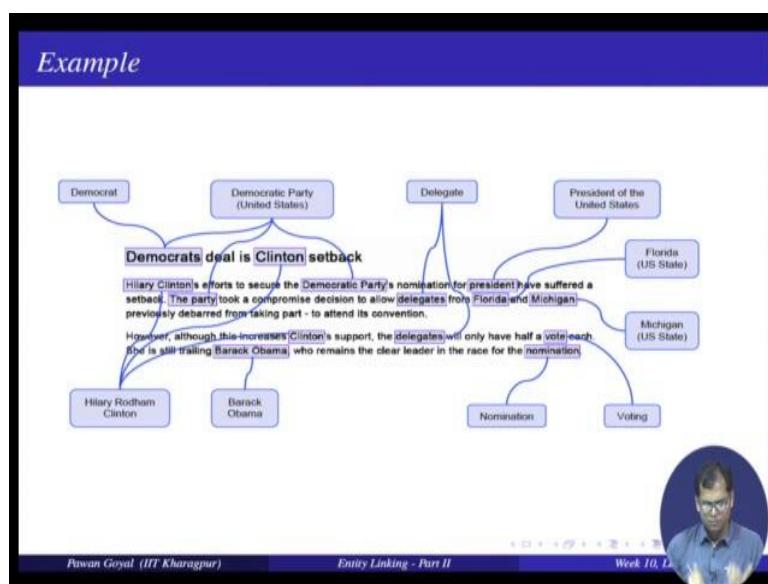
(Refer Slide Time: 15:14)



So, that is I start with a text document there I take various n-grams, it can be I take certain pattern they are noun groups or something. I take some patterns n-grams, I can say 1, 2, 3 whatever and I take some n-grams and then see what is the keyphraseness of each of these and whichever has if this is above a threshold, when I take it as a mention. And then it is a mention and then I go to my link disambiguation part, but see are you seeing that when I am finding out the appropriate entities as mentions, I am not using the context at all. I am just seeing is it a good word for, is it a good key phrase or not, does it have a good keyphraseness or not overall, so independent of a context is it a good mention or not.

So, question is can I also use the context to find out what are good mention and what are not so good mentions and that is what we will see. So, is this the best method? So, all the remaining phrases are disambiguated using the approach mentioned earlier, yes, so we have whatever mentions we have found or whatever phrases we have found we do disambiguation using an approach that we have. So, now by doing this approach you get to find a lot of different things like what are the links, what are the appropriate mentions here, what are the Wikipedia pages they link to? So you are getting some new Wikipedia pages also, now can you use this additional information to find out are they good candidates for mention at all, so that is can you use that to find out which concepts should be linked.

(Refer Slide Time: 17:14)



So, here is one example. So, you are having a Wikipedia page, so it is like a news article, so democrats deal is Clinton setback. And you are having lot of so various sentences are here.

Now, what is your approach, in your approach, you take a various mentions like Hilary Clinton occurs at various locations and try to find out what is the entity in Wikipedia it will link to. Similarly, here Barack Obama, nomination, vote, Michigan all these are link to their various Wikipedia entities. So, you get all this by your link disambiguation phrase.

Now, my question is can I use that together to find out what are good mentions also from my text. And for that we have to convert that to some sort of a learning problem. Learning problem where I run my algorithm on a data, and see what are the mentions I am detecting what are the links, I am connecting to. Now, once I have all this information, someone gives me gold standard that what are the good links here, what are not the good links, using that can I learn what are my what are the good candidates to be mentioned, what are not so good candidates to be mention. So, like coming back to my example so I start with the text data, and I find out ok, there are some mentions, they have a key phrase above a threshold.

Now, I also link them to their Wikipedia articles some of these might be linked to the same article, so that I do for this whole document. Now, suppose someone tells me that actually this is a good mention, this is a good mention, this is a good mention, but this is not so good mention, this is good mention, this is not so good mention. So, once I have all this information, can I develop a machine learning method to detect ok, given an article, given in mention and it is approved Wikipedia page all the context, is it a good mention at all. So, given a phrase with all these attributes is it a good mention for this document or not.

So, now so you can say that once you given me the text, all the all the steps that I have taken are deterministic. So, I can apply keyphraseness, I can find out the mentions, I can link them to their Wikipedia pages, so all this I can easily do. But how would I get these gold standards that this is in appropriate mention, this is not in appropriate mention and this is one of the bat bottlenecks, so how do I get this actual links and not so good links and good mentions and not so good mentions.

For that, so now what is interesting idea here, can you use Wikipedia again, so can use Wikipedia again. So, how would you use Wikipedia for this? So if you think a bit, so how you can use Wikipedia and that is actually very, very easy. So, you take Wikipedia and take some Wikipedia articles say a 1, a 2, a 3, a 4, a 5 so on. Now, each of the article now forget the hyperlink structure here, so take it as a plain text and feed it to your algorithm. So,

algorithm takes a 1 as input plain text and runs this. So, your algorithm will run this it will tell you what are the mentions, what do they link to and so on.

Now, because a 1 is already, so it is already in Wikipedia, you know what mentions are good, what mentions are not good. So, from there you can automatically construct your gold standard. And once you have the gold standard, you can apply a machine learning method to say which given a feature around this phrase, is it a good mention in this context or not, and that will solve your problem. So, this is like you are learning to link using Wikipedia. So, using the Wikipedia data and so very nicely you are taking it as a training data and also constructing your gold standard from this without having some manual efforts of labeling, because otherwise you will see this labeling will take a huge amount of time and this will help you do that automatically

(Refer Slide Time: 22:33)

The Learning Problem: Which topics should be linked?

- The automatically identified Wikipedia articles provide training instances for a classifier.
- Positive examples are the articles that were manually linked to, while negative ones are those that were not.
- Features of these articles – and the places where they were mentioned – are used to inform the classifier about which topics should and should not be linked.

Pawan Goyal (IIT Kharagpur) Entity Linking - Part II Week 10, Lecture 2 7/9

So, what will I do? So, now once I have taken the Wikipedia as input I know ok, whichever phrases gave me a Wikipedia article that was actually there in the original article, they are possible examples and whatever was not there becomes a negative example. And so you got the possible, negative examples and you feed it to your classifier. And then you use various features around these various mentions and articles to detect whether it is a good mention or not. So, you use various features like the phrases where they were mentioned to inform the classifier about which topic should and should not be linked.

So, now, what can be these possible features that you can use from a given n-gram phrase. So, let us see some features, some of these features are what you have already seen, and some other features can depend on how do people actually write a Wikipedia article. So, what are the good features? So, one feature that we can use is link probability, so that is for a given mention what is the link probability. Now, if it is occurring at multiple places like Hilary Clinton, Clinton, so they are occurring at different, different variations. So, what are they their link probabilities at each of these phrases? Take either the average or the maximum of this link probability, and that can be one feature. So, this you are doing jointly so Hilary Clinton and Clinton together, should they will linked or not. And here you are trying to use what is a link probability it at different places taking average or also taking maximum, both can be a features.

Then you can use the relatedness so how related these phrases are to the central theme of the document. So, again you will find out, what is the relatedness of these mentions with different unambiguous links in the entire article. So, this can be another feature. If they are very highly related then only you will take them as your mentions; if they are not related to the entire theme of the document; that means, they are not probably not good candidates for mentions.

Then you can also use the disambiguation confidence that is when you are trying to do a disambiguation over this mention how confident your classifier is. If your, classify is not very confident that means, you do not have sufficient context in those document and it may not be a good mention at all. So, this confidence can also be one of the features.

Then you can use the generality that is when you are trying to link some phrases in your text, so what is the idea you do not want to link something that is very, very generic that all people already know about. So, you want to link the phrases that are very specific, so how can you know about the how generic or specific a particular phrases further you can use the category tree of Wikipedia. And there you can see at what depth in the tree this particular mention comes in. So, if it comes at a very top label itself that means, it is a it is a very generic term, but this coming very low in the tree that means, say a specific term. So, specific terms might be given a high preference. So, this can also be like your feature what is the depth in the Wikipedia hierarchy tree.

And then you can also see how the documents are written that is where all this entities mentioned. So, for example, if it is a good entity, it will be mentioned in the introduction; similarly it will be mentioned in the conclusion of section of article. So, if it is mentioned in the initial few lines or the last few lines, it might be important. So, you can simply measure the offset from the beginning and the end. Then you can also see the spread that is what is the distance between then you shall mention in the last mention, so that is how far does the response across the document. If the spread is high that means, it might be a good mention; if the spread is low that means, very to only cover very small topic of this document this has been used. So, this can again be a feature for this task.

And you can think of many of other features combine these features in your classifier, and then you are learning whether given this phrase with all these features is it a good candidate for mention or not. And this is like you are learning to link using the Wikipedia structure. So, as such you take many different methods, but this is the basic conceptual idea about entity linking that how do you detect mentions different methods, once you detect mentions how do you link them to their appropriate entries in the Wikipedia or any other database, and can you use this task to also improve your mentions. And you can take it in different, different applications, take different databases, and you can try out various variations for this task. So, this so that is where we finish our discussions on entity linking.

So, in the next lecture onwards, we will start talking about information extraction that is from a document where there is a lot of unstructured data, text data, how can you identify various entities and the relations between them.

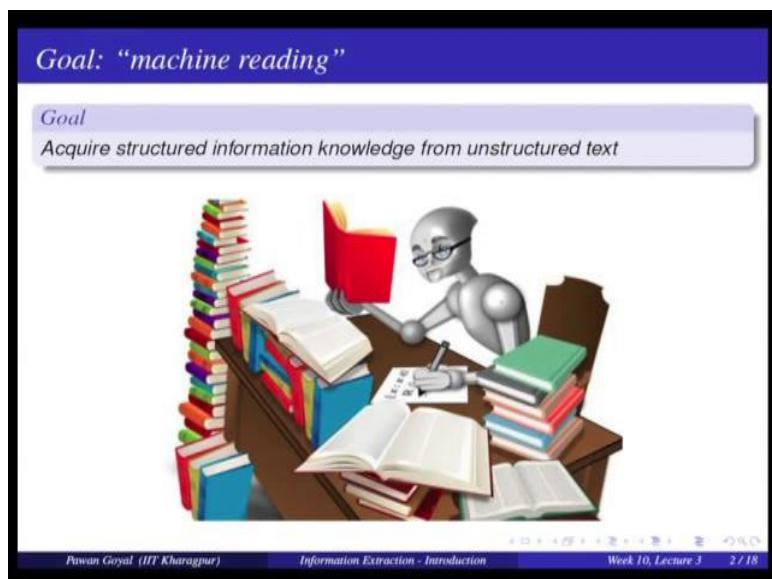
Thank you.

**Natural Language Processing**  
Prof. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 49**  
**Information Extraction – Introduction**

So welcome back for the third lecture of this week. So in this week we are doing some advanced topics on text mining. So this lecture we will start with information extraction. So we will see what are the basic, so all the basic applications where information extraction will be used, what kind of techniques you can use and we will focus our attention to a specific task that is relational extraction, how do I find out relation between any 2 entities by using the text corpus on the web.

(Refer Slide Time: 00:49)



So what do I mean by information extraction? So we can say that the goal of information extraction is like machine reading. So you have a lot of text available on the web. It is all in the unstructured form, there is no particular structure to that now from that text can I obtain some structured knowledge that can be used for various different applications and tasks. So this is like some sort of caricature to denote that yes we have a lot of knowledge available on the web and machine is trying to go through the knowledge and get some structured content that can be useful.

(Refer Slide Time: 01:29)

**Information Extraction**

**Information Extraction (IE) Systems**

- Find and understand limited relevant parts of texts
- Gather information from many pieces of text
- Produce a structured representation of relevant information:
  - Relations (in the database sense)
  - A knowledge base

**Goals**

- Organize information so that it is useful to people
- Put information in a semantically precise form that allows further inferences to be made by computer algorithms

Pawan Goyal (IIT Kharagpur)      Information Extraction - Introduction      Week 10, Lecture 3      3/18

So what the information extraction systems do? They try to find and understand various different relevant parts of the text data. So what you will have. You will have a lot of text and you are trying to get certain important information from there. So we will see what are the various ways in which you can gather this information.

So from this all this data that is available, what you want, you want to get a structured representation of some sort of relevant information. And it can be like various relations in the sense of database. So you can find out what are the various entities involved in this text and what are the relations between those entities. And can also be some sort of knowledge base that you are constructing from the data.

So the goal of information extraction is that we organize information. So that it can be useful for 2 people for doing some of their tasks. And we want to put information in a very precise form that will allow need to make further inferences. So remember this was one of the things that we are talking about in the introduction also, that the natural language text is not very precise. So how do you make, how do you convert the information to something precise that can be used for doing various task, and various inferences. And that is what we doing in the case of information extraction.

(Refer Slide Time: 03:06)

**Information Extraction (IE)**

**Definition**  
Information extraction is the task of finding structured information from unstructured or semi-structured text.

**What sort of information?**  
IE Systems extract clear, factual information

- Roughly: *Who did what to whom when?* etc.

**E.g., Gathering earnings, profits, headquarters etc, from company reports**

- The headquarters of BHP Billiton Limited, and the global headquarters of the combined BHP Billiton Group, are located in Melbourne, Australia.
- **headquarters("BHP Billiton Limited", "Melbourne, Australia")**

Pawan Goyal (IIT Kharagpur)      Information Extraction - Introduction      Week 10, Lecture 3      4 / 18

So this is a simple definition that you can get this is a sort of working definition you will say. So what is information extraction? So that is task of finding structured information from unstructured or semi structured text. So, you have the corpus or data that you have is either completely unstructured, so I can think of various treats various quora questions answers and lot of web pages' sort of unstructured. Or it can be somewhat semi structured where you have some more information like extraction information headlines etcetera, but this is not completely structured. So from this unstructured or semi structured data I want to find out a structure information, and that is what it is the definition of my information extraction.

So now, question comes in that what sort of information do you want to extract from here. So information can be something very clear and factual, like for example, this is this is something that that is normally done. So they should like who did what to whom when and who is in what relation to some other entity and so on.

So for example, suppose you have some newspaper text and they talk about various earnings profit headquarters etcetera. And this is one of the company report. The headquarters of BHP be Billiton limited and the global headquarters of the combined BHP Billiton group are located in Melbourne Australia. This is some sort of unstructured data that you can say in the form of the sentence.

Now, from this data you want to gather some structure information. So what kind of structured information do you get from this text? So you will see that, so what are the headquarters of BHP Billiton limited. So then you know the location here Melbourne Australia. And this can be some sort of structured information that you are trying to gather from the simple sentence. And that is what your information extraction system can do. So from here suppose you get this information headquarters of the BHP Billiton limited are in Melbourne Australia.

Since out of relation form there are 2 entities, and there is a relation between them and this you are extracting by using information extraction. So what is the use of doing this extraction? So once you do this extraction you will know all these tuples. So you will know these 2 entities are related are related by this relation so on and over there you can do lot of queries you can do lot of inferences and so on this can be very helpful for many question task also.

(Refer Slide Time: 05:52)

The slide has a purple header bar with the text "Information Extraction (IE)". Below it is a green box labeled "Example". Inside the green box, the text reads: "In 1998, Larry Page and Sergey Brin founded Google Inc. We can extract the following information," followed by a bulleted list: • FounderOf(Larry Page, Google Inc.), • FounderOf(Sergey Brin, Google Inc.), • FoundedIn(Google Inc., 1998). Below the green box, a blue box contains the text: "Such information can be used by search engines and database management systems to provide better services to end users." At the bottom of the slide, there is footer text: "Pawan Goyal (IIT Kharagpur)", "Information Extraction - Introduction", "Week 10, Lecture 3", and "5 / 18".

Another example let us say we have the sentence, in 1998 Larry page and Sergey Brin founded Google.

Now, from this sentence what kind of information you can get. So who are the founders of Google, and when they find when they found Google? So all this information can be extracted from here and put in a very a structured form. So like I can have a information like founder of Larry page Google founder of Sergey Brin Google and founded in

Google in 1998. So all this information is there in this text and this can be extracted by a using information extraction systems.

So now once you have this information it can be used by various search engines and database management systems to provide better services to the end users. It is not very trivial to do it directly by using the text data, but once you have this in the database form you can do a lot of different tasks and look you can use a lot of different tools to make each of this information.

(Refer Slide Time: 07:00)

The slide has a dark blue header bar with the title 'Applications in Biomedical domain'. Below the header is a white content area. A light blue rounded rectangle contains the heading 'Biomedical domain' and a bulleted list of four items. At the bottom of the slide is a dark footer bar with text and navigation icons.

*Biomedical domain*

- A large amount of scientific publications
- Need to look for discoveries related to particular genes, proteins or other biomedical entities
- Biomedical entities often have synonyms and ambiguous names
- **Critical task:** automatically identify mentions of biomedical entities in text and link them to their corresponding entries in existing knowledge bases.

Pawan Goyal (IIT Kharagpur)      Information Extraction - Introduction      Week 10, Lecture 3      6 / 18

So now what are the various applications of information extraction? For example, biomedical domain, so in biomedical domain you have a lot of research papers that are published that give details about what about the various experiments that were done using and using various patients, and what was the findings of those experiments, and what kind of drugs work what kind of drugs does not work. They can be various clinic clinical trials they can be various patrons and all that lot of information is there, but this is already very unstructured form.

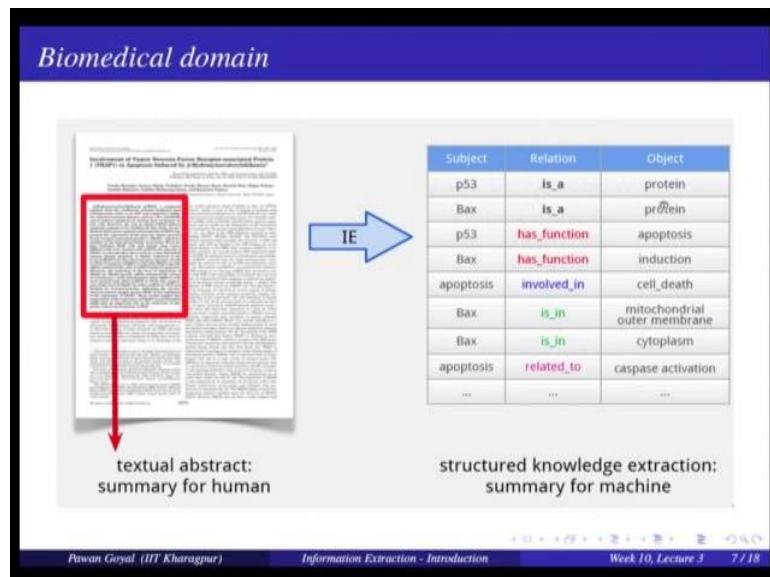
So suppose I need to look for discoveries that are related to various genes, proteins or other biomedical entities. And then the problem here could be, that these entities can have various synonyms and there are lot of ambiguities involved. So what is the task? I need to automatically identify what are the mentions of biomedical entities in the text. I

find out these are the entities that have been mentioned in the text. And then I want to link them to their corresponding entries in the lexical database.

Suppose I have a database that says these are all the different biomedical entities. Now in a research paper I need to find out this entity talks about this is corresponding to the particular entity in the database. This is very similar to the entity linking problem that we discussed in this week itself.

Now, once we find out various entities in the document, other task here could be that I want to find out how they are related to each other. So this is called relational extraction. That is one of the focus of the next 3 lectures.

(Refer Slide Time: 08:43)



So this is an example. So you have this research paper in biomedical domain and this is research paper you also get some abstract. Now from this abstract can you extract information in a structured format, like p53 is a protein, bax is a protein p53 has function of apoptosis so on. Now all this information is available in the text data, but not in this very nice structured format. So from there can you extract these are the entities and this is the relation between them.

So you find what are the entities and with different between various pairs of entities what is the relation. And this is called the structured knowledge extraction, and this is the analogy is shown here. So the research paper extract can be thought of as if something

for humans and this structured knowledge means can be thought of as something for machines. So machines can make use of this information for various tasks now.

(Refer Slide Time: 09:46)

*Relation Extraction*

CHICAGO (AP) — Citing high fuel prices, United Airlines said Friday it has increased fares by \$6 per round trip on flights to some cities also served by lower-cost carriers. **American Airlines**, a unit of **AMR**, immediately matched the move, spokesman **Tim Wagner** said. **United**, a unit of **UAL**, said the increase took effect Thursday night and applies to most routes where it competes against discount carriers, such as Chicago to Dallas and Atlanta and Denver to San Francisco, Los Angeles and New York.

Subject	Relation	Object
American Airlines	subsidiary	AMR <sup>②</sup>
Tim Wagner	employee	American Airlines
United Airlines	subsidiary	UAL

Pawan Goyal (IIT Kharagpur)      Information Extraction - Introduction      Week 10, Lecture 1

Another example: so this is like a report that you find on the web, and from this report can you extract various relations. So here you have the sentence American airlines the unit of AMR immediately matched the move a spokesman Tim Wagner said. So from here you can find out the Tim Wagner region is a spokesman for American airlines, and suppose your relation is employee. So we can say Tim Wagner is employee of American airlines, also American air airlines is the unit of AMR. So you can have this relation American airlines is a subsidiary of AMR, and similarly here united a unit of UAL you can find out this relation again.

So from this huge amount of text data can you find out this is structured information. So that is the task of information extraction. Find out the entities and what are the relations between them.

(Refer Slide Time: 10:45)

*Relation types*

For generic news text ...

Relations	Examples	Types
Affiliations	Personal Organizational Artifactual	<i>married to, mother of spokesman for, president of owns, invented, produces</i> PER → PER PER → ORG (PER   ORG) → ART
Geospatial	Proximity Directional	<i>near, on outskirts southeast of</i> LOC → LOC LOC → LOC
Part-Of	Organizational Political	<i>a unit of, parent of annexed, acquired</i> ORG → ORG GPE → GPE

Pawan Goyal (IIT Kharagpur)      Information Extraction - Introduction      Week 10, Lecture 3      9 / 18

Another example: so when the relation can be very generic. So like you can be personal relations like married to mother of organization relations like a spokesman for president of artifactual owns, something invented something, produces something they can geospatial relations, that this city is near to this city, in the on the outskirts of the city. And these kind of relations might be very helpful in replying to various queries they talk about that need geography information. See you know what cities are nearby other city. So you can try to answer these kind of questions. And directional relation this is southeast of So on. And they can be part of relations. So you need of something parent of annexed acquired for this political relation.

So you can think lots of different relations that can be established between entities. Now using these relations, you can do lot of different some sort knowledge engineering, you can do a lot of inferencing you can try to answer questions and try to predict certain relations between the entities there are lot of different tasks that you can use once do you can do once you have this structured information.

(Refer Slide Time: 12:06)

The slide has a blue header bar with the text 'Relation extraction: 5 easy methods'. The main content area contains a bulleted list of five methods:

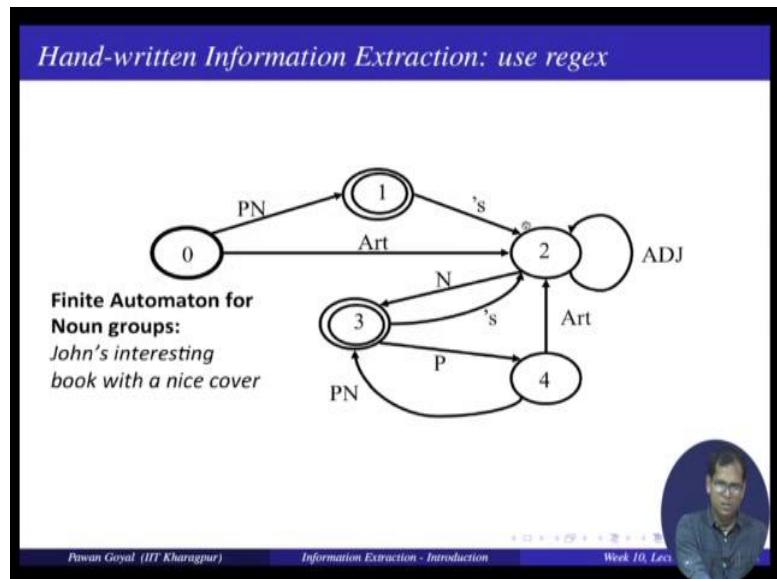
- Hand-built patterns
- Bootstrapping methods
- Supervised methods
- Distant supervision
- Unsupervised methods

At the bottom of the slide, there is a navigation bar with icons for back, forward, search, and other presentation controls. The footer of the slide includes the text 'Pawan Goyal (IIT Kharagpur)', 'Information Extraction - Introduction', 'Week 10, Lecture 3', and '10 / 18'.

So now, so the topic here is how do we gather this research information. So there are like I would say like in many different NLP applications. So here also there are 5 different methods for doing this task. So one simple method is you choose your hand built patterns. Then you get it bootstrapping methods you know supervised methods distance supervision is a very nice idea that you will see for this particular task and then you can also use some unsupervised methods.

So we will focus on the first 4 methods and we will see clearly how you can use one of these methods for the task of information extraction.

(Refer Slide Time: 12:46)



So let us see what we do in the hand built patterns. So idea is you can use various regular expressions for finding entities and the relations between them. So suppose you want to find the entities. So this is for noun groups. It is simple regular expression. So regular expression you can also denote by using a finite automaton. So here you are seeing a finite automaton that is denoting a regular expression, and this so what it is denoting any noun group.

So let us try to follow this. So you have this phrase John's interest interesting book with a nice cover. This is a noun group. So how does this automatic capture this? You say a pronoun a personal noun John, Johns interesting becomes an adjective. Book is a noun with is a preposition, article nice adjective cover noun and this is a it is a final state. So this becomes a noun group.

So we will see even John is a noun group. And John's interest interesting book is also a noun group. So it is trying to capture nouns group, noun group in various sort of granularity. You can even have single word you can have multiple words. So you it is telling you what is a noun group. Now you can further extend it to find out I know what are the noun groups now what is the relation between that.

(Refer Slide Time: 14:22)

The slide has a blue header bar with the title "Rule-based Extraction Examples". Below the header, there are three examples of entity extraction patterns:

- Determining which person holds what position in what organization**
- [person], [position] of [org]**  
Vuk Draskovic, leader of the Serbian Renewal Movement
- [org] (named, appointed, etc.) [person] Prep [office]**  
NATO appointed Wesley Clark as Commander in Chief

At the bottom of the slide, there is footer text: "Pawan Goyal (IIT Kharagpur)", "Information Extraction - Introduction", "Week 10, Lecture 3", and "12 / 18".

So suppose I want to find out which person holds what position in what organization. So what kind of patterns I can think of. A person x holding position y in organization z, so suppose I have to use some hand built patterns how will I go about it, so I will first think about what are the various kind of sentences where all these 3 entities can occur together. So there will be a person who is working in an organization. So and then once I have found some sentence is I will try to abstract what is the normal pattern that I am seeing here.

So for example one pattern can be person comma position of organization. Because you find sentences like vuk draskovic is a person comma position leader of the Serbian renewal movement. So now, what you are abstracting here, there is a person position and organization. And now you can think of many such sentences, where all these 3 entities will be there in this relation. So once you identified this pattern you will give this pattern to the machine and from there corpus it can extract all these entities for you and you will know immediately that these entities are connected by a particular relation.

What can be other patterns? So like organization named appointed etcetera person preposition office again they are all these entities. So NATO appointed Wesley Clark as commander in chief. So we are finding again all the 3 entities in a particular relation. So similarly suppose your task is to find out where is an organization located. So we will think about what are the patterns something like x located in y and or y is xs

headquarters. So we will think of these patterns and using these patterns you will try to extract these pairs of x y.

(Refer Slide Time: 16:32)

The slide has a blue header bar with the text "Rule-based Extraction Examples". Below the header, there is a pink box containing the text "Determining where an organization is located". Underneath this, there are two examples in purple boxes:

- [org] in [loc]  
NATO headquarters in Brussels
- [org] [loc] (division, branch, headquarters, etc.)  
KFOR Kosovo headquarters

At the bottom of the slide, there is a navigation bar with icons for back, forward, search, and other presentation controls. The footer contains the text "Pawan Goyal (IIT Kharagpur)", "Information Extraction - Introduction", "Week 10, Lecture 3", and "13 / 18".

Like organization location NATO headquarters in Brussels. So we will extract NATO headquarters and Brussels are the 2 entities. Organization location and you can say division branch headquarters like KFOR Kosovo headquarters. So you know this is the observation and it is a location.

So like that you can think of various patterns and extract these relations. And this is one of the very early examples on how these kind of patterns are used for extracting hyponym relation. So hyponym which is you remember, it is a relation between sub concept and a super concept. When these words first given by Hearst.

(Refer Slide Time: 17:11)

*Patterns for learning hyponyms*

*Intuition from Hearst (1992)*

Agar is a substance prepared from a mixture of red algae, such as Gelidium, for laboratory or industrial use.

- What is Gelidium?
- How do you know?

Pawan Goyal (IIT Kharagpur)      Information Extraction - Introduction      Week 10, Lecture 3      14 / 18

So what is the basic intuition? Suppose you are seeing this sentence agar is a substance prepared from a mixture of red algae such as gelidium, for laboratory or industrial use. This is sentence. Now suppose I ask what is gelidium, and you can say gelidium is some sort of algae or red algae from the sentence, yes; now how do you know that gelidium is a red alga? See you are seeing some sort of pattern here. Red algae such as Gelidium, so this pattern is telling you that gelidium is the kind of red algae.

Now, you can try to abstract these patterns, in you say that whenever you are finding such patterns x such as y, there is a hyponym-hyponym relation between x and y. And this is the idea find out many such patterns and from these patterns you try to extract these entities. So what has did, he found out various search patterns where you can have 2 entities connected by hyponym relation.

(Refer Slide Time: 18:22)

*Hearst's lexico-syntactic patterns*

*Automatic Acquisition of Hyponyms*

- $Y$  such as  $X((.,X)* (, \text{and/or}) X)$
- such  $Y$  as  $X$
- $X$  or other  $Y$
- $X$  and other  $Y$
- $Y$  including  $X$
- $Y$ , especially  $X$

Pawan Goyal (IIT Kharagpur)      Information Extraction - Introduction      Week 10, Lecture 3      15 / 18

So  $y$  such as  $x$  is for hyponyms. So what are the other kind of patterns you can use such  $y$  as  $x$  like such. Vehicle as car such vehicle as bicycle  $x$  or other  $y$  yes car or other vehicle car and other vehicle, vehicles including car and so on, vehicle especially car. So this I am given example with car and vehicles, but you can think of it as with any hyponym-hyponym pair. So he found out Freddy such titles and from these patterns he tried to extract the hyponym, hyponym relation from the text data.

(Refer Slide Time: 19:05)

*Examples of Hearst patterns*

Hearst pattern	Example occurrences
$X$ and other $Y$	...temples, treasures, and other important civic buildings.
$X$ or other $Y$	bruises, wounds, broken bones or other injuries...
$Y$ such as $X$	The bow lute, such as the Bambara ndang...
such $Y$ as $X$	...such authors as Herrick, Goldsmith, and Shakespeare,
$Y$ including $X$	...common-law countries, including Canada and England...
$Y$ , especially $X$	European countries, especially France, England, and Spain...

Pawan Goyal (IIT Kharagpur)      Information Extraction - Introduction      Week 10, Lecture 3      16 / 18

So here are some examples for these Hearst patterns and what kind of example occurrences you can see in the data. So the pattern x and other y, you can see temples tragedies and other important civic buildings. So from this sentence you can immediately see that 10 percent treasures are sub concepts of civic buildings. So we can have this pair of hyponym-hyponym. Civic build civic buildings are the hyponym and temples is the hyponym. Similarly, treasure is the hyponym x or other y. So bruises would not broken bones or other injuries. So we can have all these as a hyponym of injuries y such as x. So the bow lute such as the Bambara ndang.

So here you can see that bow lute is the super concept this is the sub concept. Such y as x such authors as Herrick goldsmith and Shakespeare. So immediately you will see there is a relation here, so on y including x y especially x.

So Hearst manually found that all these patterns, and from these patterns he was trying to extract a hyponym-hyponym pair from the data.

(Refer Slide Time: 20:17)

*Patterns for learning meronyms*

*Berland and Charniak's patterns*

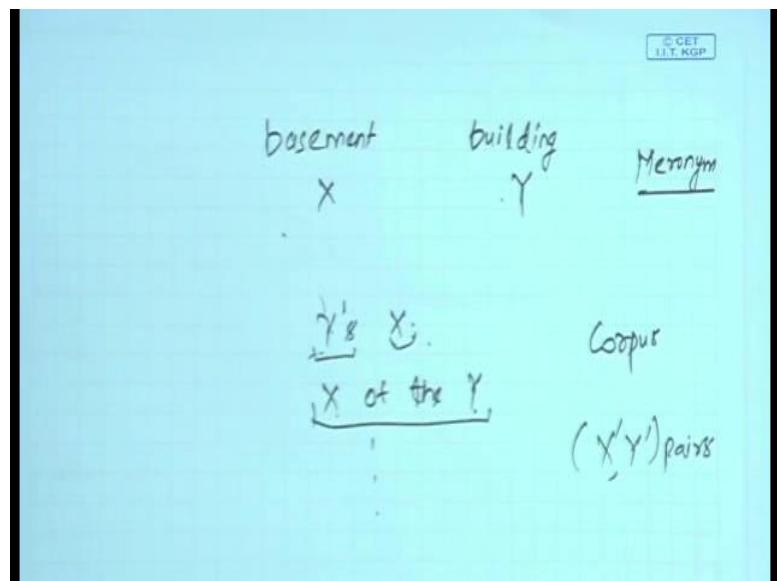
- Selected initial patterns by finding all sentences in a corpus containing *basement* and *building*

whole NN[-PL] 's POS part NN[-PL] part NN[-PL] of PREP {the   a} DET mods [ ]   NN)* whole NN part NN in PREP {the   a} DET mods [ ]   NN)* whole NN parts NN-PL of PREP wholes NN-PL parts NN-PL in PREP wholes NN-PL	... building's basement ... ... basement of a building ... ... basements in a building ... ... basements of buildings ... ... basements in buildings ...
--	--

Pawan Goyal (IIT Kharagpur)      Information Extraction - Introduction      Week 10, Lecture 3      17 / 18

Similarly, Berland and Charniaks, they found out some patterns for meronym relation, that is part of relation basement is the part of building. So they were trying to find out patterns for meronyms. So again you can think of what are the patterns that come to your mind. So like building's basement. So you will think of some example and see what kind of sentences they occurring building's basement of the building and so on and you will try to make patterns out of these.

(Refer Slide Time: 20:51)



So let us take these 2 simple examples. So like I am seeing that I have an example, basement and building. And this is my, suppose my X and this is my Y. And I want to find out many such X Y pairs that have the same meronym relation. So how will I start? I say in the sentences how will baseman in building occur together something like basements building sorry building's basement. So it will be Y's X building's basement or basement of the building X of the Y and so on and these are now my patterns. And then you will try to see in my corpus where do all these patterns occur. So example is cars wheel, wheel of the car. So we will see these X, Y are related by this meronym relation and that is how you will try to. So we are in these patterns we will try together many such X, X prime Y prime pairs.

So what Berland and Charniaks did? They selected some initial patterns for finding all sentences in the corpus that contain basement and building that is a normal is a nice method of finding these patterns. So then they found like building's basement, basement of a building basement in a building basements of buildings basements and buildings. So on now here they were writing down the patterns. So here something like NN. So they were writing in terms of what is the parts of speech that is coming and so on, so of preposition. So parts the plural noun of preposition wholes NN it is a plural noun.

So this is part in whole relation part coming as NN, in between there is the word in as a preposition or a as a determiner and in some modifiers. So there are now here abstracting.

So what they are seeing basement in a building, but it might be basement in a huge building right. So how do I absolutely better than I say there is in optionally they can be in adjective here. So that is why they are saying JJ or NN is star. Basement in a civic building and so on all these can be captured by slightly generalizing these patterns. So that is what you are seeing here JJ or NN. So you can have a civic building huge building and all this will be captured here.

So like that you try to find out these patterns and using these patterns. So once you have these patterns you try to extract some other entity pairs that are involved in this relation.

(Refer Slide Time: 23:41)

**Problems with hand-built patterns**

- Requires hand-building patterns for each relation!
  - ▶ hard to write; hard to maintain
  - ▶ there are <sup>zillions</sup> of them
  - ▶ domain-dependent
- Don't want to do this for all possible relations!
- Plus, we'd like better accuracy
  - ▶ Hearst: 66% accuracy on hyponym extraction
  - ▶ Berland and Charniak: 55% accuracy on meronyms

Pawan Goyal (IIT Kharagpur)      Information Extraction - Introduction      Week 10, Lecture 3      18 / 18

So this is a nice method if you want to sit down and look at each and every relation and think about the patterns. And that is also the problem with this approach that some persons who are good with the data, who are also language they know how the system work, they can they can try to get you some hand built patterns. So now, question the problem is that they are hard to write and hard to maintain and there are like you can think of zillions of patterns. So we can think of so many different ways in which people can talk about hyponym-hyponym pair in the data.

So how do I capture all of these patterns manually and yeah there might be domain dependent. So every domain you might have different ways of writing things, and yes you can do that for some relations, but suppose they are thousands of relations. How do you do for all these thousand relations? So we and these patterns that Hearst found or

Berland and Charniaks found they were giving kind of results, but they were not like giving very accurate results. So for example, Hearst patterns give the roughly 66 percent accuracy on hyponym extraction, and Berland and Charmian gave 55 percent accuracy on meronyms.

So we would like probably prefer to have better accuracy than these numbers. So that is using hand built patterns you can only go little. So only small distance and there are also a lot of manual effort is required. So how can we avoid this manual effort? And that is what we will see in the other approaches, that we will be discussed it starting from, how do we do simple bootstrapping here, and that is all you will be discussing in the next lecture.

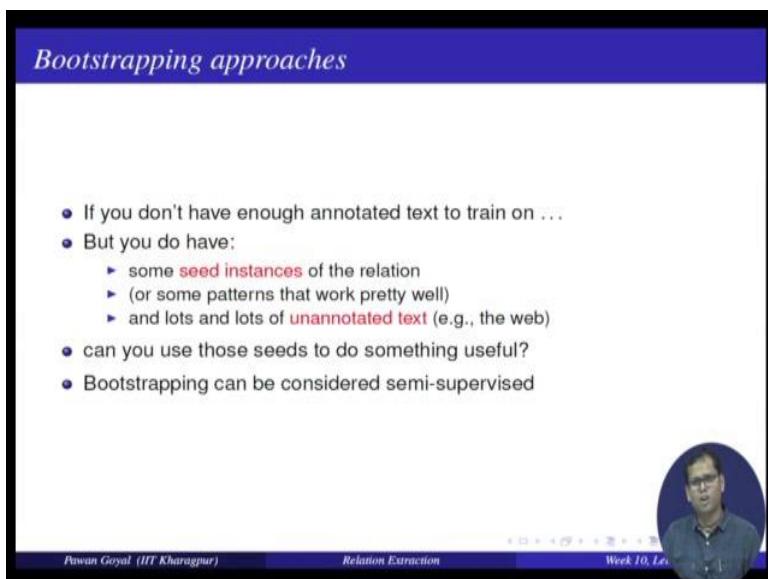
Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 50**  
**Relation Extraction**

Welcome back for the 4th lecture of this week. So, we started talking about information extraction and there we discussed about the particular problem of relation extraction and we discussed some so, one approach that is using hand built patterns. So, we can build some patterns manually and we saw that by using that we can extract certain relations between entities. And there were some limitations with that. So, we will now see some other approaches apart from hand built patterns.

(Refer Slide Time: 00:46)



The slide has a blue header bar with the text "Bootstrapping approaches". The main content area contains a bulleted list:

- If you don't have enough annotated text to train on ...
- But you do have:
  - ▶ some **seed instances** of the relation
  - ▶ (or some patterns that work pretty well)
  - ▶ and lots and lots of **unannotated text** (e.g., the web)
- can you use those seeds to do something useful?
- Bootstrapping can be considered semi-supervised

At the bottom of the slide, there is a video player interface with a circular video thumbnail showing a person's face, the text "Pawan Goyal (IIT Kharagpur)", "Relation Extraction", and "Week 10, Lec 50".

Starting with the bootstrapping approaches, so we had talked about bootstrapping approaches in one of the earlier topic that is on word sense disambiguation, but let us see how do we apply these approaches for the task of relation extraction.

Here you will use these approaches only if you do not have a lot of annotated data because if you have some annotations already available then you can use some supervisor approaches that might give you better results, but suppose you do not have annotated data; that means, data where it is labeled entity 1 is related to entity 2 by a particular relation R, if you do not have such data you will use your bootstrapping approach.

You do not have enough data, but what you have are some seed instances of the relation and that is very easy. Now what do I mean by seed instances? So, remember we were talking about hyponyms or meronyms, see all you know some seed relation. So, you know basement and buildings are connected by the meronym relation, car and vehicles are connected by the hyponym relation.

You know some seed instances then what else do you need? Or you might have some patterns that you know and lots and lots of unannotated data; that means, you should have a lot of corpus where you have a lot of text data it may be an unannotated, it is a simple text data. So, idea is using some seed patterns, running some idea over this whole data you can try to bootstrap your approach for relation extraction.

The questions here are, so suppose so here you have some seed instances. So, how do you use them for doing something meaningful or so, that you can extract where are other entities? So, you have some seed instances how do you use that for extracting more such examples and so, this can be consider some sort of a semi supervised approach. So, what do we do here? So, let us take some simple example.

(Refer Slide Time: 02:55)

### Bootstrapping example

- Target relation: burial place
- Seed tuple : [ *Mark Twain*, *Elmira* ]
- Google for "Mark Twain" and "Elmira"
 

"Mark Twain is buried in Elmira, NY."  
   → X is buried in Y  
 "The grave of Mark Twain is in Elmira"  
   → The grave of X is in Y  
 "Elmira is Mark Twain's final resting place"  
   → Y is X's final resting place
- Use those patterns to search for new tuples

Navigation icons: back, forward, search, etc.

Pawan Goyal (IIT Kharagpur)      Relation Extraction      Week 10, Lecture 4      3 / 17

Here suppose I have my relation is burial place. So, X is buried in place Y, I want to find out all such entities that are connected by this relation.

How do I go about it? Firstly, I need to see if I have some seed instances that is some entities that are connected by this relation suppose I have an instance that is Mark Twain is and the burial place is Elmira, New York. So, I have this seed tuple, now how do I start? So, remember what did we need? We need some seed instances and a lot of corpus data.

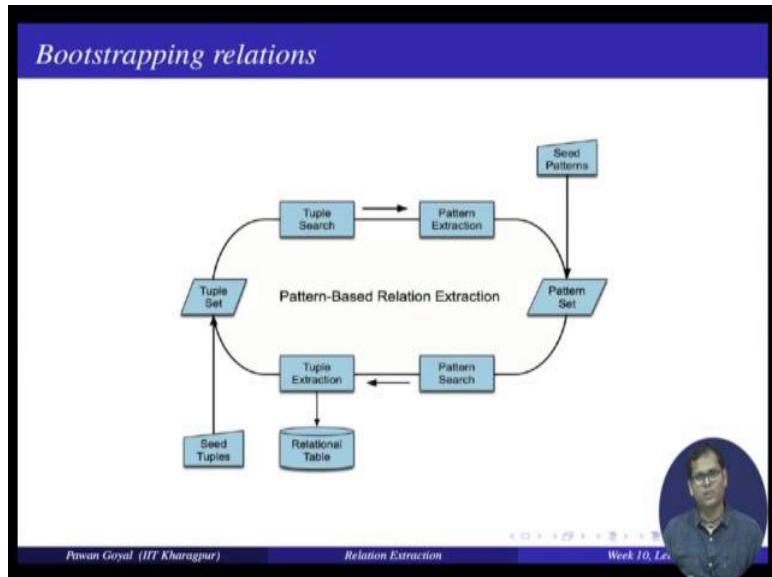
Now you can use the intuition that how are you finding out the patterns in the case of hand built patterns. So, again we were starting with some hand with some seed instance. So, like basement building and we were trying to go to the corpus and seeing wherever these 2 words occur what is the pattern in which they occurring, same thing you can do here. So, you know I have 2 entities Elmira and Mark Twain. Now you can search your whole corpus to find out where all these 2 entities occurred together any single sentence or in some proximity.

Now, wherever they occur you try to find out in what pattern do they connect to each other and these patterns you can generalize as your gen generic patterns without having to manually go through each of these? So, something like so, I have these and it is Mark Twain, Elmira and maybe I can google this google this google this entities Mark Twain Elmira together and let us see, what do I; what are the kind of sentences do I get and suppose I get a sentences like this, excuse me, Mark Twain is buried in Elmira New York, this is the sentence and I know what am I entities here Mark Twain is X Elmira is Y. So, I can immediately from a pattern X is buried in Y in this become my pattern.

The next sentence the grave of Mark Twain is in Elmira. So, I can have this pattern the grave of X is in Y similarly Elmira is mark twins final resting place. So, I have this pattern now Y is X S final resting place and so on. So, we are getting this sentence is from the sentence you are extracting over you are entities X and Y and you are building patterns in terms of X and Y. So, what you saw here just by using 1 seed instance and a lot of unannotated data you can find out some patterns.

Now, once you have patterns, what you will do? You will use these patterns to find out more and more such entity pairs that are connected by this pattern and that will enhance your seed tuples or seed instances then you can again use this seed instances to again such the corpus find out more such patterns and this can be done in an iterative manner until there is some convergence going on or you are seeing that there is not helping much.

(Refer Slide Time: 06:06)



Now, you have these patterns and you use these to search for new tuples. So, describe described by the simple flow chart that is you are starting with this some sort of seed tuples like Mark Twain Elmira then you are searching with tuples in the corpus and you are finding various patterns and this becomes your pattern set you might also have some seed patterns already, but you are getting some patterns. Now using this patterns you are searching the corpus and finding more tuples and I putting them in your relational table and that is going to your tuple set, now using your tuple set you can search these and find out more patterns and this can keep on going in many many iterations and this is in a very nice approach you can see that you only need data that is freely available everywhere and you need some very few seed instances and you can apply this algorithm and this does not require to do everything manually.

(Refer Slide Time: 07:01)

The slide has a blue header bar with the title "Bootstrapping problems". The main content area contains a bulleted list of problems:

- Requires that we have seeds for each relation
  - ▶ Sensitive to original set of seeds
- Generally have lots of parameters to be tuned
- No probabilistic interpretation
  - ▶ Hard to know how confident to be in each result

At the bottom of the slide, there is a video player interface showing a man speaking. The video player includes the following labels: "Ptwan Goyal (IIT Kharagpur)" on the left, "Relation Extraction" in the center, and "Week 10, Lecture 1" on the right. There are also standard presentation navigation icons.

Yes, but there are some problems with that approach the problems. For example, are that the seed instance with which we start should be a good instance such that there are many occurrences of this seed in the corpus if the seed pair does not arrange the corpus, you will not be able to extract many relations many patterns from this. So, this is one problem with this approach. So, this is sensitive to the original set of seeds that you use for your algorithm and in general there can be many parameters to be tuned, for example, how many top patterns will I take from my set? How many iterations I will go through and how many times I will send the same pattern for my first search? There can be many a many parameters that you have to fix and yeah there is no such probabilistic interpretation. So, it is difficult to know how confident you are in each pattern or each tuple that you are finding by this approach.

Here some sort of problems with this approach, but it is a nice approach if you use to want to get it done without building some sort of machine learning method also, so on, you can use this approach very for some simple and easy results, but we will see if you want to build a more over system that what kind of approaches you can you can use.

(Refer Slide Time: 08:21)

The slide has a blue header bar with the title "Supervised Relation Extraction". The main content area contains a bulleted list of steps:

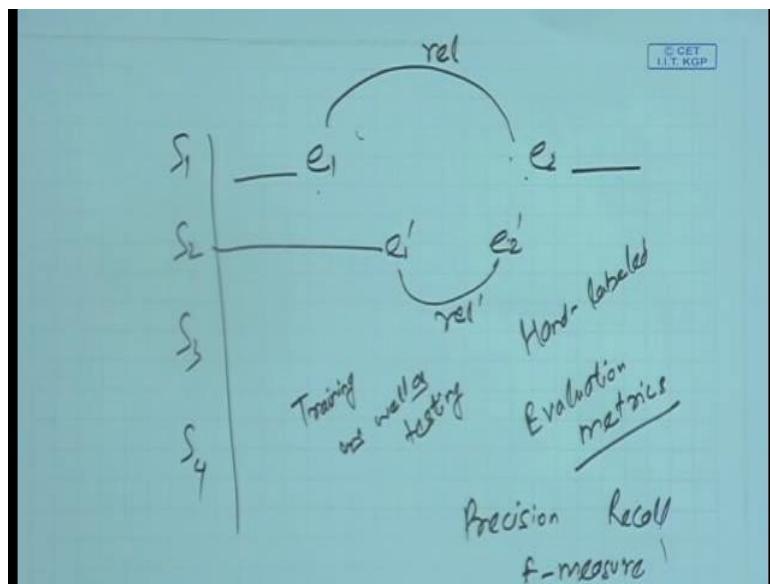
- Choose a set of relations you would like to extract
- Find and label data
  - ▶ Choose a representative corpus
  - ▶ Label the named entities in the corpus
  - ▶ Hand-label the relations between these entities
  - ▶ Break into training, development and test
- Train a classifier on the training set

At the bottom of the slide, there is a footer bar with the following information:  
Ptwan Goyal (IIT Kharagpur)      Relation Extraction      Week 10, Lecture 4      6 / 17

For that we can talk about some supervised approaches for relation extraction. So, what is the idea? So, first you will define, what are the kinds of relations you want to extract? So, relations can be many like I want extract family relations. So, parent of wife of husband of and so on, you can extract some organization relation, this is an employee of and subsidiary of and so on. So, we will define a set of relations.

Now, for each relation, you will find data and label the data. So, they should be some manual labeling involved somebody has to label the data that in this sentence these entities are connected by this relation. So, you will choose a representative corpus where you think that there can be some instances of this relation now you will label the named entities in the corpus and hand label the relations between the entities.

(Refer Slide Time: 09:25)



What will happen? You have a corpus you will find out sentence  $S_1, S_2, S_3, S_4$  and you say in the sentence, this entity 1, this entity 2 and you know what is the relation between them, similarly here you find there is 1 entity; 1 prime entity, 2 prime and there is some relation prime here and this has to be hand labeled why do you need hand labeling. So, once you have these hand labels they are like your; so this is like your bold standard that you can use as your training as well as testing data.

We will train your system using in this sentence, if these are the entities there is a relation between them. So, in a new sentence suppose 2 entities are there is there is the relation between them. So, this can be some machine learning model that you can built by using this gold standard and then you will yeah break into training development in text that is the usual practice in machine learning and then you will train a classifier on the training set.

(Refer Slide Time: 10:39)

*Supervised Relation Extraction: An extra step helps*

- Find all pairs of named entities (usually in same sentence)
- **Extra step:** Build a binary classifier to decide if 2 entities are related
- If yes, use another classifier to classify the relation

*Why the extra step?*

- Faster classification training by eliminating most pairs
- Can use distinct feature-sets appropriate for each task

Pawan Goyal (IIT Kharagpur)      Relation Extraction      Week 10, Lecture 4      7 / 17

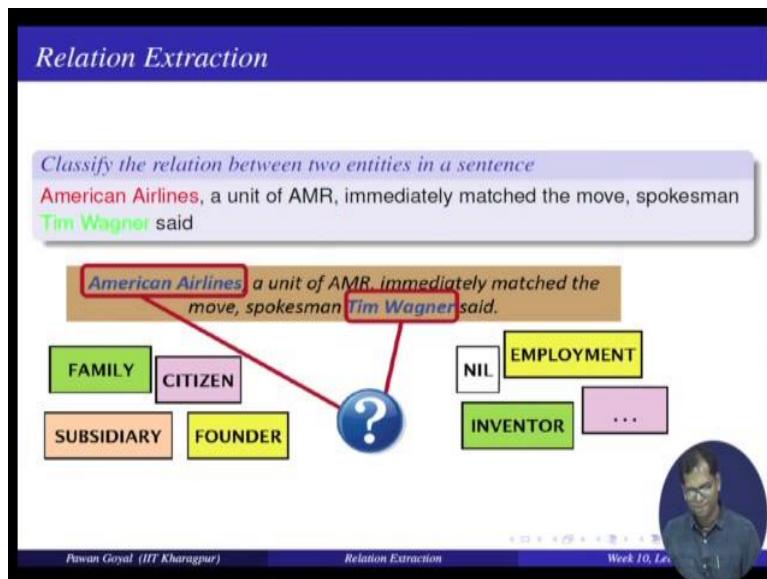
Now so here it might be one problem while you are using this approach in general, they can be 100s of relations and so you need to; so when in you are given a sentence between 2 entities take they can be many relations, but they may not be any relation at all, it might happen that 2 entities are occurring in a sentence, but they are not connected by any relation (Refer Time: 11:04) there is no relation as such.

What you might do is to have the first step that says in a sentence, I know what are all the entities and this first step tells me which 2 entities are connected and which 2 are not connected and once you have the output of this step, you know these 2 entities are connected then you run your additional classifier to find out, what is the relation between them among all the 100s of relations that you have.

This issue; this is seemed to be working in this in this area of relation extraction that first you find out what entities are connected second what is relation between them. So, it is a 2 stage approach. So, you find all pairs of named entities in a sentence and the extra step can build a simple binary classifier. So, that is says yes or no and it decides whether 2 entities are related or not and if you find an answer, yes, to this is step use another classifier to find out what is the relation between these 2 entities and why will that help because in the first step build itself you will be able to eliminate a lot of extra pairs that are not involved in any relation. So, we will not bother about those you will only bother about those entities that are probably connected by some relation.

Other advantage could be you can think of the idea sort of features that you will used for the first step that is finding out if there is a relation or not and the second step that is if this relation what is that relation you can think of very set of features that you can imply both for both of these steps.

(Refer Slide Time: 12:46)



Here is one visualization of what this will look like. So, suppose you have this sentence American Airlines a unit of AMR immediately matched the move a spokesman Tim Wagner said. So, now, suppose by using the first step you found out that American Airlines and Tim Wagner are connected by a relation. So, now, you want to find out, what is the relation? So, here you have a sentence, 2 entities you need to find out, what is the relation among all these possibilities is family relation, citizen relation, subsidiary relation, founder relation and so on. So, lots of relations are there you want to find out what is the relation between these 2 entities.

Now, how do you solve this problem? How would you solve this problem? So, you will have a lot of labeled data when you know these are the entities here and this is the relation between them. So, in classification, what we do from this labeled data? We try to abstract over some sort of features. So, we will say so, these are the features that I see in this sentence and these features indicate relation 1, other sentence I am seeing this kind of features that indicating relation 2 and so on. So, this is what I will have from my training data. Now attached data again, I will try to find out what are the features and using these features I will try to match

with one of this previous examples I have seen in training data, this is simple illustration, but it is generally more complex than that, but this is the basic idea.

So, the whole effort goes in deciding, what should be my ideal features by which I can represent all my data points. So, how do I say, these 2 are connected by this relation? What are the different things in the surrounding, in the context about these entities that I should be using to make this decision? And that is your task of each engineering find out what are the features that will help you in this task.

In most of the NLP application, this is one of the main challenges that for this task find out, what are the appropriate sets of features I can use. So, we will see some examples at what? So here you can use all the different concepts that are covered in this course. So, it is starting from simple language models part of speech tags dependency parse synthetic parse everything, you can use to do find out to define, what are your features in this task and you will see a lot of examples here and this is one. So, if you want to build your own system you might have to start thinking in terms of what are the important insights from data that I here use as in the form of my features?

Remember, features are something that you think can help me discriminate between various relations here. So, it can help me, tell me tell if this is a family relation versus if it is a citizen relation, what are the different things that can help are these various words that occur in the context are these part of speech tags and or this is a something else. So, this you can abstract in terms of here features.

(Refer Slide Time: 15:54)

**Features: words in mentions M1 and M2**

American Airlines, a unit of AMR, immediately matched the move, spokesman Tim Wagner said.

*Bag-of-words features*

WM1 = {American, Airlines}, WM2 = {Tim, Wagner}

*Head-word features*

HM1 = Airlines, HM2 = Wagner, HM12 = Airlines+Wagner

Pawan Goyal (IIT Kharagpur) Relation Extraction Week 10, Lecture 4 9 / 17

Let us see what are the kinds of features, we can use for this task. So, I have the sentence American Airlines etcetera and my initial features could be what are the words in my mention; M 1 and M 2? So, M 1 is American Airlines M 2 is Tim Wagner. So, what are the words that I used in these 2 mentions? So, feature here can be bag of words features. So, mention 1 uses word like American Airlines and mention 2 uses words like Tim Wagner. So, these are simple features I can also use what are the headwords of these 2 mentions the headword mention of mention 1 is airlines and for mention 2, it is Wagner and you can also see, what is the headword mention of 1 plus 2 airlines plus Wagner?

Why you are using this headword kind of features? So, here you are having American Airlines, but suppose there is something like Indian airlines or some other airlines. So, by using the headwords using capture E 1, a new word that has the same headword, but the initial word was different it can be captured by using headwords same with Tim Wagner. So, you are capturing the surname here by headword, but suppose if someone else has a surname Wagner. So, it can also be used.

(Refer Slide Time: 17:11)

American Airlines, a unit of AMR, immediately matched the move, spokesman Tim Wagner said.

Words or bigrams in particular positions left and right of M1/M2  
M2:-1 = spokesman, M2: +1 = said

Bag of words or bigrams between the two entities  
{a, AMR, of, immediately, matched, move, spokesman, the, unit}

Ptwan Goyal (IIT Kharagpur) Relation Extraction Week 10, Lecture 4 10 / 17

Then I can use, what are the words that are coming around the mentions? So, that is what are the words are coming before American Airlines after Tim Wagner and what are the words in between? So, what can be my features? So, words or bigrams in particular positions left and right of M 1, M 2 like what is the word before M 2. So, it is a spokesman, what is the word next to M 2? That is said, so what you are abstracting here? I have an entity before which I have a word spokesman and next word is said. So, the new context whenever I see, what is spokesman before, what said afterwards, it might indicate that there might be this relation, a spokesman X said it might be a good indicator of this relation.

You can also use the back of words or bigrams between the 2 entities that is what are the different kind of words that occur between the 2 entities here? So, we will say. So, words like AMR immediately matched a spokesman unit etcetera they are all occurring between the 2 entities and they all go as your features.

(Refer Slide Time: 18:18)

**Named Entity Type and Mention Level Features**

American Airlines, a unit of AMR, immediately matched the move, spokesman Tim Wagner said.

*Named-entity types*  
M1-NE = ORG, M2-NE = PERSON

*Concatenation of the two named-entity types*  
M12-NE = ORG-PERSON

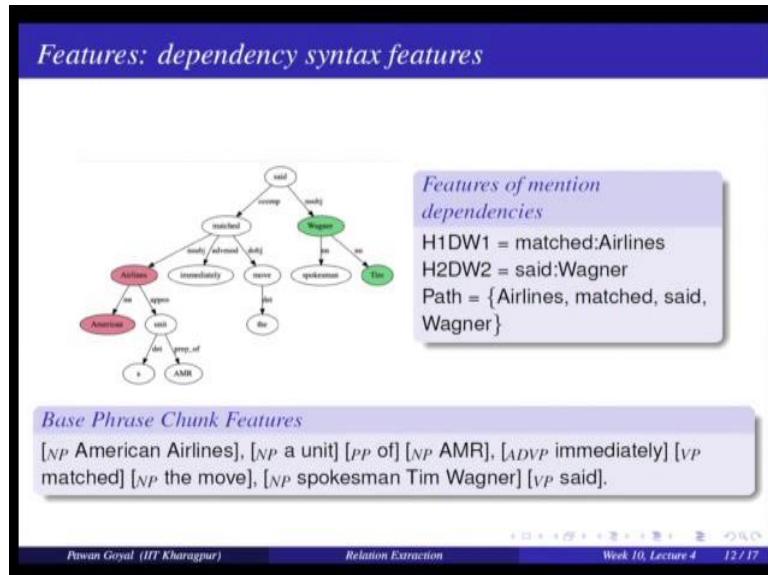
*Entity Level of mentions (Name, Nominal, Pronoun)*  
M1:EL = Name, M2:EL = Name  
'it' or 'he' would be pronoun, 'the company' would be nominal

Pawan Goyal (IIT Kharagpur)      Relation Extraction      Week 10, Lecture 4      11 / 17

You can have named entity type and mention type features. So, for example, what is the named entity tag for the mention one? So, it is like American Airlines organization. So, this you can get by using various named entity recognition tools you can run in any area and you can find out what are the various named entities. So, it is say mention 1 is in organization.

Similarly, mention 2 is a person. So, this can be nice feature that can help you, this is an organization, this is a person. So, what can be a relation between them and yeah it can be together also, what is the named entity for 1 in 2 together organization person then you can also find out entity levels of mentioned is the name nominal or pronoun. So, here first one is a name second one is also a name, but suppose in the sentence you have it he etcetera. So, we can call it as a pronoun on the other hand if you have a word like the company you will call it as a nominal. So, all these can also be your features.

(Refer Slide Time: 19:25)

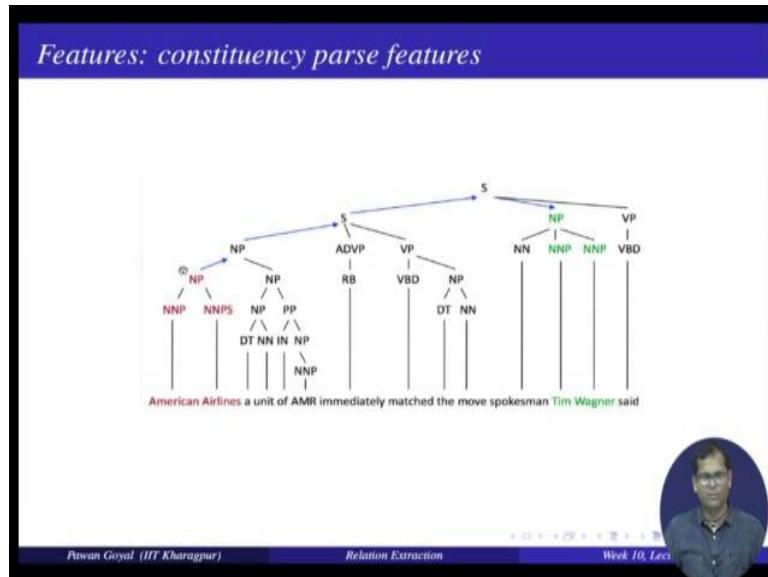


Now, suppose you want to use the dependency between them. So, you will see when you convert the sentence to a dependency graph, what is the connection between the 2 entities? What are the different branches in that in the tree by which they are connected? So, suppose you find this dependency graph, so we will say ok. So, they are connected by this path matched, said and you, are saying going to Wagner airlines matched, said, Wagner. So, now, you will try to use certain features based on this path also. So, it can be maybe what are the words that occurring in this path or what is the complete path altogether?

Here, what is the headword of the dependence; the dependency headword for word 1. So, you had airlines matched airlines is the dependency for the word one for headword 2. So, we have the dependency said and Wagner this can be a feature immediate dependency feature matched airlines said Wagner then what is the path airlines matched said Wagner and you can also think of some other features what is the label they are at in that dependency graph how many different words they are connected to and so on.

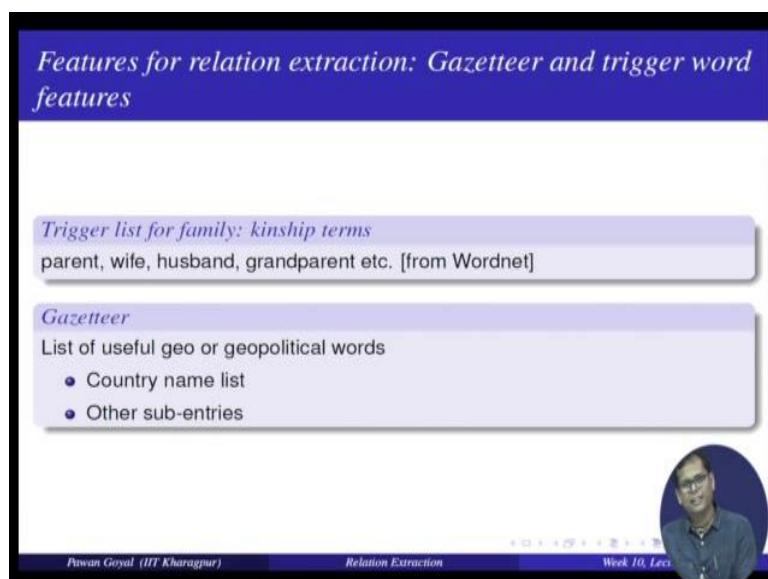
And then you can also do chunking and is use used features like if you chunk them you will find American Airlines a unit of AMR etcetera and your feature could be what is the chunk in which it participates what is the next chunk after this and so on.

(Refer Slide Time: 20:58)



You can also use the constituency parse feature. So, you have the 2 words here, American Airlines and Tim Wagner and this is the party of the sentence. So, you can use the path from here for noun phrase to this particular noun phrase that connects Tim Wagner. So, what is the path here going to an noun phrase to a sentence to a sentence to a noun phrase this path can be helpful again here you can use what is the label in the tree they are at and what is the sibling and so on. So, these can be your various features.

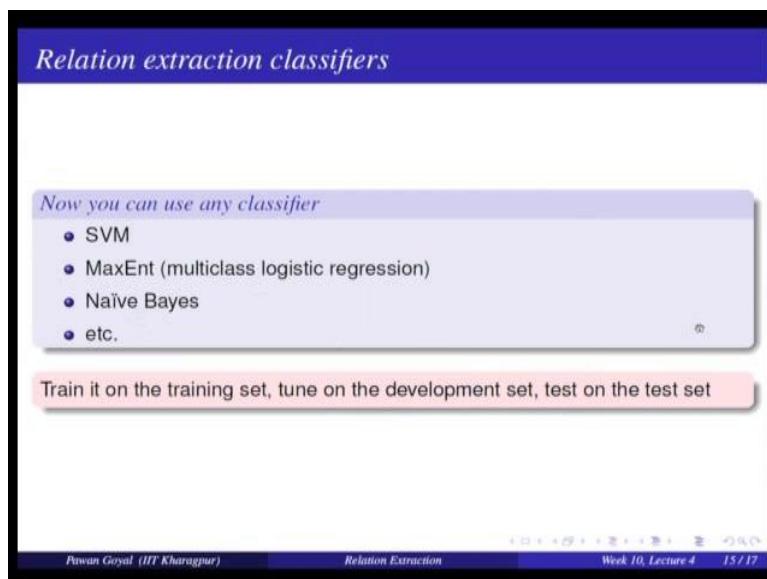
(Refer Slide Time: 21:32)



Then they can be some sort of features that you can obtain from various gazetteer and different terms kinship terms. So, if your relations contain mother of and parents of an all that. So, you can use some kinship terms. So, like parents wife husband grandparent etcetera and this can be obtained from various electrical sources like Wordnet also and then you can use various gazetteers like, what are the country name lists? So, you can see.

The next word after the entity 1 is a name of the country or the previous word after the entity 2, any of a country, similarly for names of very famous celebrities or persons all these can be used as your features. So, this is the idea that that you have this task you have to find out the relations and you should be able to use whatever sort of features you think will help in this task. So, we are seeing we are using a lot of different sort of features and so and they are using all the different concepts and topics that you have seen in the basics of this course.

(Refer Slide Time: 22:44)



You can have country name list others of entries etcetera. So, now, once you have identified what are your features, so what will happen if you are training data, each instance you can convert in a feature vector. So, we know what are the different features that are involved in this particular sense and then you have you can use various classifiers to built to find out given a new sentence how do I find out if the 2 entities have a relation and what is the relation and then you can you can use multiple classifiers like naive Bayes classifier or SVM or MaxEnt etcetera and this is the rule always you train on the training set tune on development set and test from the test sets.

You should never use your test set for building your features or whatever patterns. So, you should never use it as set you should be kept separate. So, you will only tune on your development set if you your training set you run your classifier and initially test on the development set if it does not work keep on improving and once you are satisfied then the on the test set and find the accuracy and you might also have to compare with others if you want to find out if you are doing something better than what people have already done.

(Refer Slide Time: 24:01)

*Evaluation of Supervised Relation Extraction*

Compute  $P/R/F_1$  for each relation

$$P = \frac{\text{Number of correctly extracted relations}}{\text{Total number of extracted relations}}$$

$$R = \frac{\text{Number of correctly extracted relations}}{\text{Total number of gold relations}}$$

$$F_1 = \frac{2PR}{P+R}$$

Navigation icons: back, forward, search, etc.

Bottom footer: Piwan Goyal (IIT Kharagpur), Relation Extraction, Week 10, Lecture 4, 16 / 17

$P = \text{Number of correctly extracted relations} / \text{Total number of extracted relations}$

$R = \text{Number of correctly extracted relations} / \text{Total number of gold relations}$

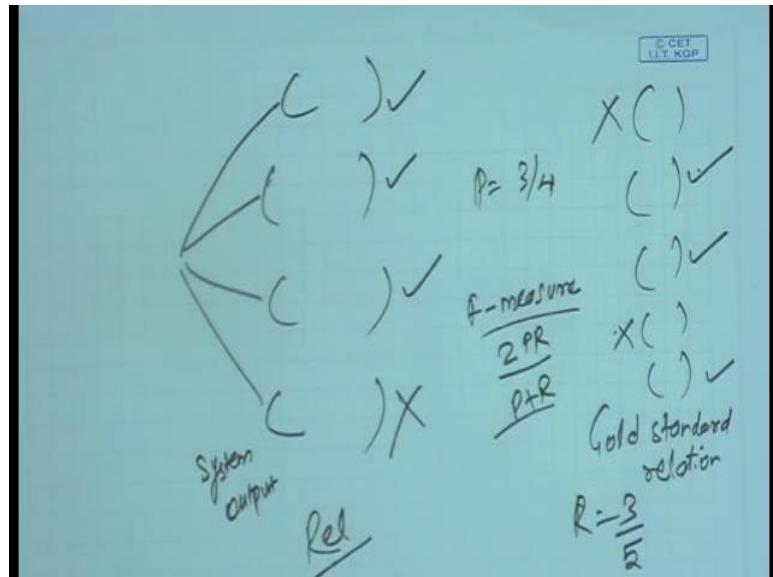
$$F_1 = 2PR/(P + R)$$

So, now how do I, so once you have done this classifier, you will get your; so classifier will tag in this sentence these 2 entities are connected by this relation and so on. Now how do you find out how good your system is performing and how we compare with the system. So, for that we will talk about what are the various evaluation measures or evaluation metrics matrix, how do I evaluate it? So, in standard that evaluation metrics are, what is the precision? What is the recall and what is the F major?

So, how do I define these? What is the precision recall etcetera? So, here are the simple definitions. So, precision is so for suppose, I am doing it for each relation separately. So,

precision would be what are the number of correctly classified or extracted relations divided by total number of extracted relations?

(Refer Slide Time: 25:03)



That is suppose my system is giving, so these entities these pair of entities are connected by a relation R, this is the output of my system.

Now, from a gold standard, suppose I know that this and this is correct and this is incorrect, system will give 4 output; 3 are correct. So, precision here will be 3 by 4. So, this is one very important metric that my system should have a high precision; it should be whatever relational predicting should be correct, what is the other criteria? So, the criteria is recall is what are the number of correctly extracted relations, but my system identified divided by total number of gold relations, now it is important to find the see the distinction between that 2.

So, what we will do in re recall? So, the system output in requires the see what is the what are the gold standard relations supposing my gold standard I had 5 entities I know these entities are connected by this relation and suppose my system has found out. So, it has found out 3 it is found out this, this and this, but my system could not find out this and this. So, what is the recall of my system my system could recall 3 out of 5 possible relation. So, recall here would be 3 by 5 and this I can do for every relation independently and I can show my system does work for this relation with this precision this recall and so on.

Now, there is also a metric where you can combine these 2 and that is called f measure and what you do is to take some sort of harmonic means. So, this is  $2 P R$  divided by  $P + R$ , you will take a harmonic mean of precision recall and that is called your F measure. So, this is  $F \text{ measure} = \frac{2 P R}{P + R}$ , although there are variations where you can give different weights to precision and recall also, but this is quite accepted measure that you give equal weightage to precision recall and find out an F 1 measure.

(Refer Slide Time: 27:24)

**Supervised RE : summary**

Supervised approach can achieve high accuracy	But has significant limitations!
<ul style="list-style-type: none"> <li>At least, for some relations</li> <li>If we have lots of hand-labeled training data</li> </ul>	<ul style="list-style-type: none"> <li>Labeling large training set (+ named entities) is expensive</li> <li>Doesn't generalize to different relations</li> </ul>

Navigation icons at the bottom right include back, forward, search, and other presentation controls.

Now if we try to summarize the supervised relations extraction task, what we did here? So, in general, it can achieve very high accuracy for some relation and if we have lots of hand labeled training data. So, for most of the machine learning algorithms, they all depend on how much label data that you have. So, if you have lots and lots of data, they can give you better, better and better accuracy. So, that is one bottle neck also. So, you need to label lot of data to be able to get good accuracy and. So, this is the limitations here. So, labeling large training set and the entities might be very very expensive and it may not generalize to different relation. So, I have labeled for some relations, but suppose I want to now extract some new relation I cannot use this label, I need to get new labels for the new relation

Whatever I have labeled, the model will only be able to extract those relations a new relation I have to do the labeling again. So, this also does not generalize. So, we sought to approaches. in this lecture, we saw bootstrapping, you have you do not want to annotated lot of data here, some seed instance you go for that, but you can if you can annotated lot of data then you can

use a supervised approaches and they are the main problem is after labeling find out interesting features that can help with this task.

Now, in the final lecture for this week, you will see an interesting approach that blends these 2 approaches together; bootstrapping and supervised approaches and this can also generalized to many many different relations that that you have. So, this is take this approach is called distance super provision. So, in the next lecture, we will talk about that approach.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 51**  
**Distant Supervision**

Welcome back for the final lecture of this week. So, for relation extraction, we have talked about hand built patterns, we have talked about bootstrapping approaches and also supervised approaches. So, in this lecture, we will talk about a very interesting approach that uses the previous 2 approaches nicely, it is called distant supervision and we will see what is the basic idea hoping you will find it very interesting so what is the idea?

(Refer Slide Time: 00:42)

**Distant supervision paradigm**

**Hypothesis**  
If two entities belong to a certain relation, any sentence containing those two entities is likely to express that relation

**Key Idea**  
Use a database of relations to get lots of training examples

- instead of hand-crafting a few seed tuples (bootstrapping)
- instead of using hand-labeled corpus (supervised)

**Approach**  
For each pair of entities in a large database:

- Grab sentences containing these entities from a corpus
- Extract lots of noisy features from the sentences
  - Lexical features, syntactic features, named entity tags
- Combine in a classifier

Pawan Goyal (IIT Kharagpur)      Distant Supervision      Week 10, Lecture 5      2 / 22

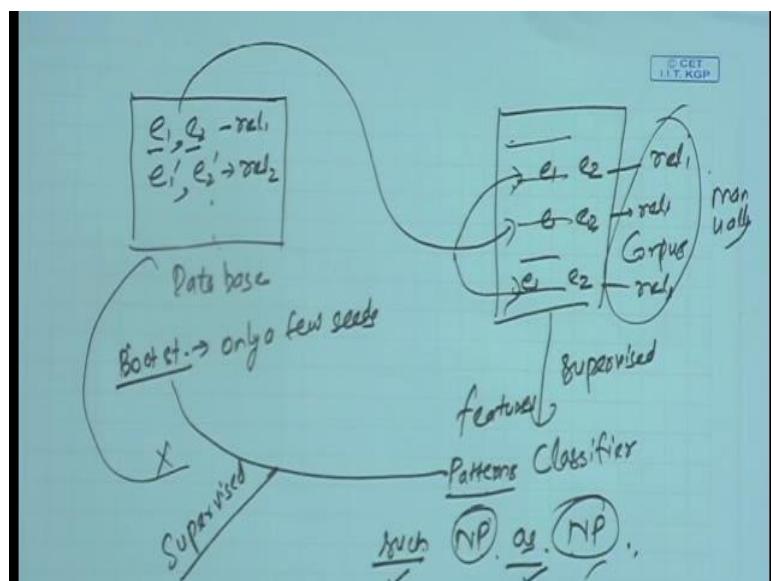
We start with the hypothesis that is similar to what you were using also in bootstrapping. So, what is the hypothesis? If there are 2 entities that are connected by a particular relation, whatever sentence in the corpus they occur in that sentence conveys that relation that is the hypothesis that is what this model is built on that I have 2 entities, I know there is relation between them if you find a sentence in the corpus where these entities are there, I can think of these are connected by that relation only. So, yeah it is likely to express that relation. So, what is the key idea here? So, we are trying to make use of both bootstrapping and supervised ideas. So, that is so here what we do? We start by using a database where you already have lot of instances of a particular relation. So, lot of seed tuples are there.

You use database of relations to get lots of training examples, instead of handcrafting a few seed tuples that you do in bootstrapping in bootstrapping you take only probably few seed tuples and you start doing some iterative method in or and instead of using hand labeled corpus that we do in the case of supervised approach. So, you do not do either of this, but you start with a database that contains lot of substitution examples.

So, what is approach? So, you have an database where you know what are the entities that are connected by different relations now you automatically grab various entity pairs from this database and you will lot of unannotated data you go through that corpus and find out wherever each entities occur you try to extract various features from there you do not take simple patterns you grab them you know you take them as the labeled sentences extract features from that.

You extract grab sentences containing these entities from a corpus extract lots of noisy features from the sentences these can be lexical features syntactic features named entity tags this is what you were doing in supervised approach, but now you are doing that without any hand labeling and then combine this in a classifier.

(Refer Slide Time: 03:13)



If we try to understand that from a block diagram, what we will do? So, we will have a database that will contain entity 1, entity 2 and the relation 1 entity. 1 prime entity. 2 prime relations 2 and so on and this is what you will have in the database then you will have a lot of corpus there are lot of sentences. So, what you will do you will use the hypothesis to say that

wherever these entities are found in this corpus they are connected by this relation pi one. So, here you have the entity 1, entity 2, entity 1, entity 2, entity 1, entity 2, in these sentences.

Now, you start assuming that this is your labeled data set. So, now, you know this is your labeled data set. So, you will say you will label it by the label one or by this hypothesis this is relation one this is relation one this is your labeled data set now you do exactly what you do using supervised approach. So, we will try to use this is my relations now I will extract features from here sorry. So, that is what words come before what words come after what is the parse tree everything you can use. So, you use these features to build the classifier and that classifier when it runs over new sentences can give you labels these 2 entities are connected by this relation and so on.

So, what is so now, also we can see how it is different from bootstrapping, in bootstrapping you start with only a few seeds, now you go to the corpus, here you do not take the features in bootstrapping, you only take the patterns simple patterns and here not as good as using the features in a classifier and what you do in the case of supervised approach in supervised approach this has to be done manually you have to manually tag each sentence with the relation entities with the relation and this you are avoiding because by using database. So, this database is not there in the case of supervised approach. So, you directly go to the corpus you start labeling and use those labels to create the classifier.

So what? So, you see here trying to combine bootstrapping and supervised in a nice way such that you are avoiding the problems with each of these you are avoiding the problem with bootstrapping that is no problems with interpretation you do not know how that will do good or bad there are only a few seed tuples by taking a lot of examples; and doing with features instead of patterns and you are avoiding problem with supervised approaches by avoiding your labeling task and taking from database the labels. So, that is how you combine both of these approaches together.

(Refer Slide Time: 06:33)

**Benefits of distant supervision**

*Has advantages of supervised approach*

- leverage rich, reliable hand-crafted knowledge
- relations have canonical names
- can use rich features (e.g. syntactic features)

*Has advantages of unsupervised approach*

- leverage unlimited amounts of text data
- allows for very large number of weak features
- not sensitive to training corpus: genre independent

Pawan Goyal (IIT Kharagpur) Distant Supervision Week 10, Lecture 5 3 / 22

Now, let us say so this has advantage of supervised approach by leveraging rich handcrafted knowledge and you can use these features and also of the unsupervised approach because you can leverage on unlimited amount of text data, it may not been labeled then also you can use it in this approach and you can use a lot of different weak features. So, syntactic features lexical features and so on and you know it will not be sensitive to the training corpus. So, it will work on whatever corpus you are you are trying to use it on.

(Refer Slide Time: 07:06)

**Hypernyms via distant supervision**

Construct a noisy training set consisting of occurrences from a corpus, that contain hyponym-hypernym pair from Wordnet.  
Ex: **Shakespeare - author**

*Training yields high-signal examples like:*

- "...consider **authors** like **Shakespeare**..."
- "Some **authors** (including **Shakespeare**)..."
- "**Shakespeare** was the **author** of several..."
- "**Shakespeare, author** of **The Tempest**..."

*But also noisy examples like:*

- "The **author** of **Shakespeare** in Love..."
- "...**authors** at the **Shakespeare** Festival..."

Pawan Goyal (IIT Kharagpur) Distant Supervision Week 10, Lecture 5 4 / 22

Let us take an example of finding Hypernyms via this distant supervision idea. So, here so what we will do? So, what is a database which will contain the Hypernym pairs and you can see wordnet is the database that contains super concept sub concept in a nice machine readable manner. So, I will try to use that database to find out more Hypernyms. So, let us take an example by a simple single Hypernym suppose one example is Shakespeare and author Shakespeare is an author. So, what I will do similar to bootstrapping I will go through my corpus and find out where all these entities occur together in a sentence.

Suppose I find out some sentences like this consider authors like Shakespeare some authors including Shakespeare, Shakespeare was the author of several Shakespeare the author of temptation so on so, these are the very sentences where I am finding these entities and I will immediately label them with the relation of hypernym that entity one Shakespeare here related to entity 2 author by this relation of hypernym. So, this labeling can be done automatically now because I have these relations and the hypothesis.

Now, one problem here might be that there are some noisy examples like the author of Shakespeare in love authors at the Shakespeare festival. So, can you see why this is noisy? So, author and Shakespeare both words occurring together, but they are not occurring in the relation of hypernym you see I am talking about here the author of a book Shakespeare in love it is not talking about Shakespeare being an author similarly here some authors at a festival we are not talking about Shakespeare being an author. So, what will happen suppose you take it as your pattern authors of Shakespeare or authors at the Shakespeare festival if you take it as your pattern if you try to extract new entities they will they will not be correct. So, think about a pair of words that occur with this relation or this pattern.

This can be someone was at somewhere. So, it does not say that this is sub concept super concept relation, no. So, this is your noisy example now how will my distant supervision handle these noisy examples. So, in the case of bootstrapping there is no easy way to handle these examples, but here what I will do because I have many many seed pairs for this relation. So, this noisy example will appear in may be one or 2 cases, but they will not occur in most of the cases and that is what I will make use of. So, they will not they will occur may be 1 or 2; 2 hypernym; hypernym pairs, but they are not occurring with many other pairs and if I take random pairs they might occur with them. So, so that is why these noisy examples will not deteriorate the performance of my system.

(Refer Slide Time: 10:14)

The slide has a blue header bar with the title 'Learning hypernym patterns'. The main content area contains a bulleted list of steps:

- Take corpus sentence  
... doubly heavy hydrogen atom called deuterium ...
- Collect noun pairs  
e.g. (atom, deuterium)  
752,311 pairs from 6M sentences of newswire
- Is pair an IS-A in WordNet?  
14, 387 yes; 737, 924 no
- Parse the sentences
- Extract patterns
- Train classifier on patterns  
logistic regression with 70K features

At the bottom of the slide, there is footer text: 'Ptwan Goyal (IIT Kharagpur)', 'Distant Supervision', 'Week 10, Lecture 5', and '5 / 22'.

How do I learn the hypernym patterns? I take corpus sentence like doubly heavy nitro hydrogen atom called deuterium and I have the entities like various nouns like atom deuterium etcetera. So, so if I take. So, here what was done 6 million sentences were taken from newswire and 752311 pairs were formed of entities noun pairs. So, this is starting from the data. So, you have a lot of data from that you are finding noun pairs and you have found you are seeing the 750000 noun pairs.

Now, to use distant supervision you have to see some of these noun pairs are already involved in hypernym relation. So, you can use wordnet to find out how many of these are having this relation. So, suppose from there you find out fourteen thousand roughly are having this relation and remaining seven thirty seven thousand do not have this relation. So, immediately you have the positive examples and the negative examples from the data.

Wherever one of these 15000 entities occurs, you will say these are connected by the relation of hypernym and wherever these entities occur they are there is no hypernym relation between these entities and this you can use to build your classifier and extract various features. So, you parse the sentence extract patterns build various features and train classifier on these patterns and suppose, in this experiment people so they run logistic regression classifier with 70000 different features that was considered using patterns and everything else.

(Refer Slide Time: 12:08)

### Syntactic dependency paths

Patterns are based on paths through dependency parses generated by MINIPAR. Example word pair: (Shakespeare, author)  
Example sentence: "Shakespeare was the author of several plays..."

*Minipar parse:*

Extract shortest path:  
-N:s:VBE, be, VBE:pred:N

Pawan Goyal (IIT Kharagpur) Distant Supervision Week 10, Lecture 5 6 / 22

Now patterns are so in this approach what the patterns? They used were based on a specific dependency parse it is called Minipar parse. So, so this will tell you how you can find out nice patterns and use them as your features also and. So, what is the idea for using Minipar for considering your patterns as features? So, suppose we have the sentence Shakespeare was the author of several plays and you want to find out a feature between the 2 entities Shakespeare and author you want to find a feature between these 2 entities.

What you will do? You will run this parser; this Minipar parser and you will get the parse of this sentence. So, you have this parse of the sentence. So, you have you have the lemmas here Shakespeare author of and be and they are connected by various relations different words are connected now you want to extract a relation between or a pattern that connects Shakespeare and author. So, what is the idea you will abstract towards the words that are that you want to connect because you want to find out what are the other words that are connected by the same sort of pattern. So, you will extract over Shakespeare and author call them x and y.

You are starting from Shakespeare. So, you will see what is the path that connects the 2? And you are going in the opposite direction here. So, you are seeing, what is this path? What is the first dependency relation that connects Shakespeare and in Minipar? What happens, you write down what is the part of speech of the first word, the relation and part of speech of the next word that is how the relations are labeled the relations are labeled.

Here Shakespeare is a noun, be is a verb and the relation is subject and you are going in the opposite direction. So, you are having a minus. So, you denoting this relation by minus noun subject and VBE, VBE is the part of speech which tags for be then you will take the intermediate words. So, you will take the word be here then you have another path similar again take v b e predicate and the noun and you will stop here because you want to connect Shakespeare and author. So, by this simple approach you can try to find out patterns between any 2 possible words from the dependency parser.

What they did? They found out all the possible patterns between different words in the corpus and so, there was 70 k; 70 k different patterns that the users feature for training their dependency curve for sorry not dependency curve, but their logistic regression for finding out if 2 words have a relation of hypernym.

(Refer Slide Time: 15:02)

The slide has a blue header bar with the text "Syntactic dependency paths". The main content area contains a bulleted list of rules:

- Original nouns in the noun pair are removed to create a more general pattern
- Each dependency path is presented as an ordered list of dependency tuples
- Optional "satellite links" are added to each shortest path  
"such NP as NP" : function word 'such' is added to the shortest dependency path

At the bottom of the slide, there is a circular profile picture of a man, likely the speaker. The footer contains the text "Pawan Goyal (IIT Kharagpur)", "Distance Supervision", and "Week 10, Lec 1".

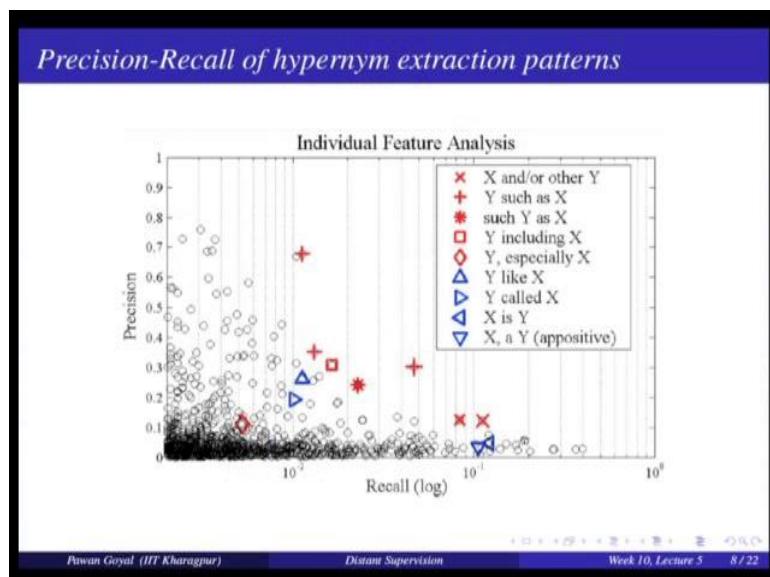
Now, what was so and they did some more tricks like as we discussed original nouns in the noun pair were removed to create a more general pattern and each dependency path is presented in ordered list of dependency tuples that you were seeing and sometimes they have some satellite links like such and as.

What is found when you are building patterns function words are very important to the patterns, but the content words are not so important. So, wherever you have a pattern where you have a word like such there is a noun phrase as noun phrase. So, what you will do? You will retain the function words, but the noun phrases you will put them as noun phrases instead

of the word. So, instead of saying such red algae as *Gelidium*, you will say such NP as NP and such and as can remain as it is because these function words are very helpful in describing your pattern and this is a nice trick that you can use for building your patterns. So, abstract over the content words and retain the function words.

Even content words you might take some content words that are having very high frequency because then you might find them more often, but if something is not having high frequency there is no point in keeping the word as it is better to abstract, it as a part of speech tag or the particular phrase and so on and this helps a lot in building these patterns and this you can use as your features or whatever.

(Refer Slide Time: 16:39)



Now once they did that so this in this figure, what you are seeing? Individual feature analysis, so if you take only one feature at a time what is the precision recall that you get on your data and you are having. So, these are like seventy k different features and you are seeing most of the features had very poor precision recall independently, but there are some features that is stood out that were giving very high precision recall and when they analyzed they found something very interesting.

These are the features that were having nice recall in precision and these are the features X and the other Y is such as X such Y as X Y including X Y especially X now do you remember these features. So, when we talked about the patterns that Hearst felt manually he was using these features. So, so these features you may not be identified in this approach we

never gave these features manually, but the algorithm was able to identify these features automatically what were the some other features like Y like X Y called X, X is Y X A Y. So, these were also some other features that Hearst did not find, but algorithm was finding and if you show them to various linguists or to various people they will say oh yes these features make sense for this particular task. So, this was very very interesting about this approach.

(Refer Slide Time: 18:08)

The slide has a blue header bar with the text "What about other relations". Below it is a white content area. At the top left of the content area, there is a small icon of a person and the text "Distant supervision". In the center, there is a table comparing two datasets:

Training set	Corpus
 Freebase	 WIKIPEDIA
102 relations	1.8 million articles
940,000 entities	25.7 million sentences
1.8 million instances	

At the bottom of the slide, there is a footer bar with the text "Piwan Goyal (IIT Kharagpur)", "Distant Supervision", "Week 10, Lecture 5", and "9 / 22".

Now this was just for a single relation hypernym but suppose you want to do that for many many different relations at the same. So, together I want to train them together how will I do that. So, that is where in 2 thousand nine this was this paper by the Jurafskys group distant supervision for relation extraction without labeled data. So, what they did in place of the corpus they used Wikipedia. So, that had at that time 1.8 million articles and 25.7 million sentences that was there unlabeled data set what was their database containing the relations they used freebase now freebase is a very large repository of various entity pairs and their relations. So, that time it contained 102 relations 940000 entities and 1.8 million instances that many different pairs of entities are present with their labeled relations.

(Refer Slide Time: 19:04)

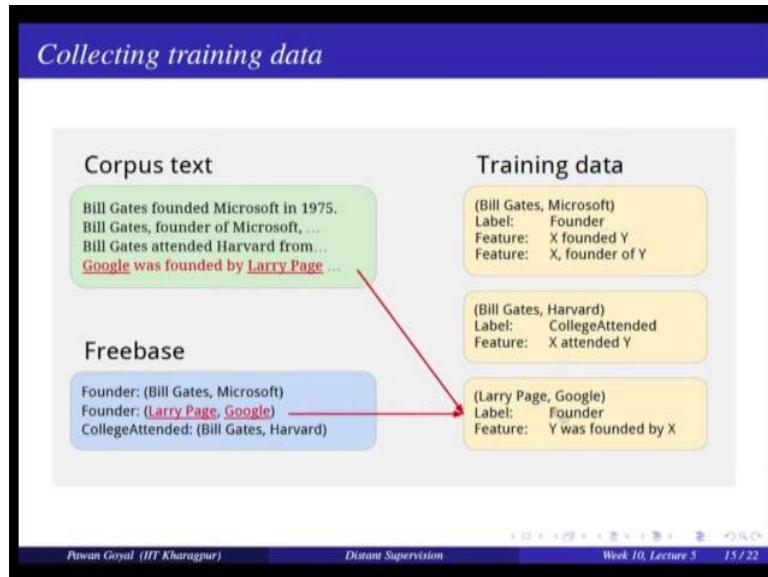
Frequent Freebase relations		
Relation name	Size	Example
/people/person/nationality	281,107	John Dugard, South Africa
/location/location/contains	253,223	Belgium, Nijlen
/people/person/profession	208,888	Dusa McDuff, Mathematician
/people/person/place_of_birth	105,799	Edwin Hubble, Marshfield
/dining/restaurant/cuisine	66,529	MacAyo's Mexican Kitchen, Mexican
/business/business_chain/location	42,806	Apple Inc., Apple Inc., South Park, NC
/biology/organism/classification.rank	40,658	Sacculinaria, Order
/film/film/language	31,103	Where the Sidewalk Ends, Film noir
/biology/organism_higher_classification	30,052	Enter the Phoenix, Cartonidae
/film/film/country	27,217	Calepteryx, Calopterygidae
/film/writer/film	23,856	Irving Shulman, Rebel Without a Cause
/film/director/film	23,539	Michael Mann, Collateral
/film/producer/film	22,079	Diane Eskenazi, Aladdin
/people/deceased_person/place_of_death	18,814	John W. Kern, Asheville
/music/artist/origin	18,619	The Octopus Project, Austin
/people/person/religion	17,582	Joseph Chitttrand, Catholicism
/book/author/works_written	17,278	Paul Auster, Travels in the Scriptorium
/soccer/football_position/players	17,244	Midfielder, Chen Tao
/people/deceased_person/cause_of_death	16,709	Richard Dainree, Tuberculosis
/book/book/genre	16,431	Pony Soldiers, Science fiction
/film/film/music	14,070	Stavisky, Stephen Sondheim
/business/company/industry	13,805	ATS Medical, Health care



Ptwan Goyal (IIT Kharagpur)      Distant Supervision      Week 10. Lec 1

Now, how does relations look like in freebase? So, you will have a relation name. So, in the category of people there is a person and nationality that is how you need to read it there are 2 eighty one thousand relations and 1 example John Dugard is the person with nationality South Africa, similarly location contains Belgium, contains Nijlen person profession Mister McDuff is mathematician person, place of birth, Edwin Hubble born in Marshfield. So, like that this freebase contains all these relations and what are the entities that are connected by these relations. So now so what is their task using these whole data sets try to come up with the algorithm classifier such that you can populate this database so that you can find out new and new entity pairs that are connected by relations. So, what is the idea? What they do?

(Refer Slide Time: 20:00)



Here is the simple illustration. So, you have this corpus text like Wikipedia. So, there are various sentences that are here then you have your freebase that contains the relation and entities that are connected by the relation like founder Bill Gates founded Microsoft, Larry Page founded Google, Bill Gates attended Harvard. So, various relations and the entities, now how do you make use of both of these? You will take; you will either take the relations here and go through the corpus and find out wherever they occur together in the single sentence. So, you will say Bill Gates and Microsoft occur together in this sentence. So, you will say; that means, a Bill Gates and Microsoft are the entity pairs they occur in the sentence. So, I gave a label as founder and then I extract various features.

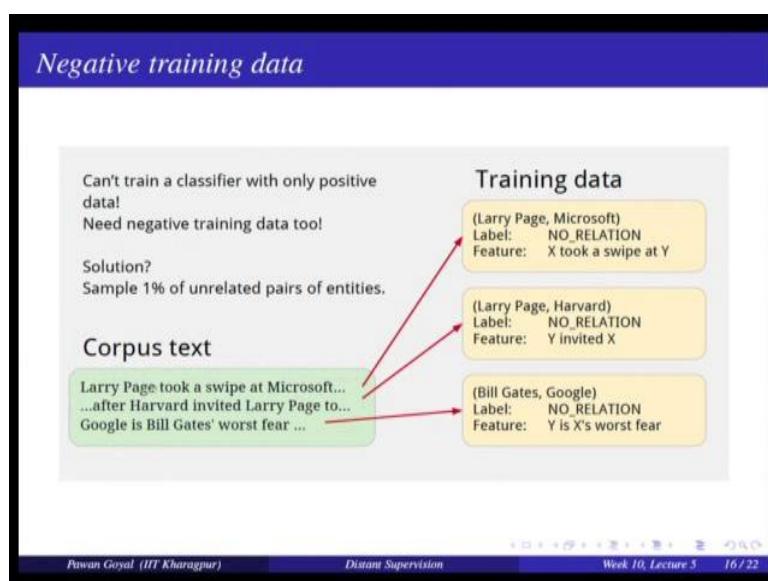
Here for simplicity we are only showing one simple pattern X founded by, but in general you can think of any other feature like what is the next word to Y it is in and what, what occurs between X and Y is X starting the sentence. So, like that you can have many many different features. So, you are giving a very simple feature here X founded Y as the pattern. So, so that is what you are doing for each entity pair you are going to corpus selecting sentences where they occur using the hypothesis they might be connected by the same relation labeling relation and extracting features.

Another sentence it occurs, so you will extract features X founder of Y now suppose you go to Bill Gates, Harvard, you have this sentence and you will extract the feature X attended by relation college attended similarly here Larry Page Google. So, founder Y was founded by X.

So, by different sentences you will try to you will be able to get different sort of features and different sort of patterns now these are all positive examples of various relations you might also have to get some negative examples that these 2 entities are never connected by any relation now how.

It is interesting that how do you construct negative examples in this distant supervision approach and the idea is also very interesting that you have your database you know what entities are connected, but from your database also sample some pair of entities that are not connected by any relation. So, once you sample these entities find out the sentences in the in the corpus where these are appearing together and label this as no relation because we know there is no relation in the in the freebase among these entities and this will generate your negative data. So, this is your positive data and then you will get some negative data also. So, you cannot really classify with only positive data. So, you need some negative trained data.

(Refer Slide Time: 22:57)



What you do? Sample 1 percent of unrelated pairs of entities, 1 percent because otherwise there will be too much negative data. So, you do not want to make a classifier where it is more biased towards the negative examples. So, we take some sample you under sample the negative cases and here is how you will do that.

Suppose you find out in your corpus or sorry in your database there is no relation between Larry Page and Microsoft. So, you will find out the sentence like Larry Page took a

swipe at Microsoft and you will say with this feature the label is no relation X took a swipe at Y no relation here Harvard invited Larry Page to something. So, Y invited X no relation and so on we will do for all these pair of entities.

(Refer Slide Time: 23:51)

The slide has a blue header bar with the text 'Preparing test data'. Below it is a main content area with a light gray background. On the left, under 'Corpus text', there is a green box containing the sentence: 'Henry Ford founded Ford Motor Co. in... Ford Motor Co. was founded by Henry Ford... Steve Jobs attended Reed College from...'. An arrow points from this text box to a yellow box labeled 'Test data'. This 'Test data' box contains two entries:

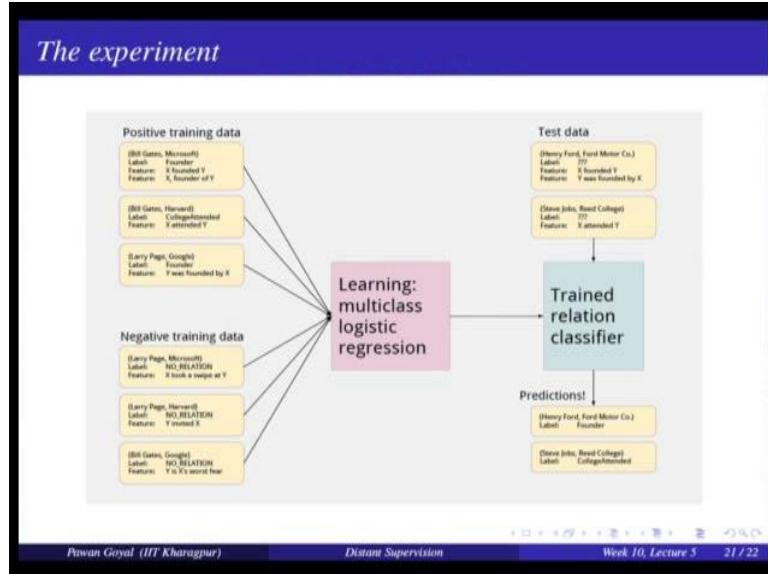
- (Henry Ford, Ford Motor Co.)  
Label: ???  
Feature: X founded Y  
Feature: Y was founded by X
- (Steve Jobs, Reed College)  
Label: ???  
Feature: X attended Y

At the bottom of the slide, there is a dark footer bar with the text 'Piwan Goyal (IIT Kharagpur)', 'Distant Supervision', 'Week 10, Lecture 5', and '20 / 22'.

You have now got and see here what will happen in the testing? Testing, you will have the corpus text. So, you will have the sentence here you find out the entities and then you will parse it through your classifier to find out is there a relation between these 2.

You have the feature like X founded Y for the entities, but you did not know the label in the text side. So, you will use the classifier to find out with this feature is there a relation between them and if. So, what is the relation and you will say similarly here Y was founded by X, what will be the label X attended Y? What will be the label and the classifier would be able to tell this will be college attended this will be founder of and so on.

(Refer Slide Time: 24:37)



In a single figure if you want to see so we will have some positive training data you will have some negative training data you will use this to learn your logistic regression and here is a multi class where no relation is also a class and you have many different relations at test time what will happen for each sentence you will take the entity pairs you will try to identify the features and you will feed it to your classifier to find out with these features what is the relation between these 2 entities.

Here suppose the predictions are Henry Ford and Ford motor company, the label is founder Steve Jobs and Reed College, the label is college attended, now what are the kind of features you can use? So, all these we have talked about the features buts.

(Refer Slide Time: 25:23)

**Features**

Each feature describes how two entities are related in a sentence, using either syntactic or non-syntactic information.

**Lexical Features**

- The sequence of words between the two entities
- The POS tags of these words
- A window of  $k$  words to the left of Entity 1 and their POS tags
- A window of  $k$  words to the right of Entity 2 and their POS tags

**Feature conjunction**

- Each lexical feature consists of the conjunction of all these components
- A conjunctive feature is generated for each  $k \in \{0, 1, 2\}$

Pawan Goyal (IIT Kharagpur) Distant Supervision Week 10, Lecture 5 22 / 22

But you can use features like. So, they should tell how the 2 entities are related in the sentence. So, you can use lexical features in terms of only the words you can use syntactic features in terms of what their connections you can use dependency paths like the example we discussed you can use gazetteers named entities lot of different things that you can make use of.

Lexical features can be like what are the sequence of words between the 2 entities parts of speech tags of these words a window of  $k$  words of the left of entity one their part of speech tags a window of  $k$  words to the right of entity 2 their part of speech tags and you can also combine these features I will take part of speech tag of the previous word. And the part of speech tag of the next word I can also try to combine these features and generate various conjunctive features and then use a use lot of syntactic features and once you identified these features use that to train your regression classifier and get the output.

That is where we will end this week. So, we talked about some interesting application on entity linking and information extraction relation extraction and we saw that they are nice ways in which you can make use of lot of data that is available on the web and put it in a more usable manner, so entity linking so that you know given an entity what particular concept it corresponds to in a database in information extraction you can find out given entities - what is a relation in which they are what is the relation that connects these 2 entities and you can populate a knowledge base using that and these are some of the very nice and

important applications used in text mining. So with that we end this week and I will then see you in the next week.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 52**  
**Text Summarization – LEXRANK**

Hello everyone, welcome to the week 11 of this course. So, we were talking about various applications and this week we will focus on two different applications; text summarization and text classification. So, this lecture we will start talking about text classification and we will focus on one very important approach that is using LEXRANK algorithm for summarization.

(Refer Slide Time: 00:45)

The slide has a blue header bar with the text "Text Summarization". Below it are three main content boxes:

- What is a summary?**

A summary is a text that is produced from one or more texts, that contains a significant portion of the information in the original text(s), and that is no longer than half of the original text(s). (Hovy, 2008)
- What is text summarization?**

Text summarization is the process of distilling the most important information from a source (or sources) to produce an abridged version for a particular user or task. (Mani and Maybury, 2001)
- Humans have an incredible capacity to condense information down to the critical bit.  
"He said he is against it."  
Calvin Coolidge, on being asked what a clergyman preaching on sin said.

At the bottom of the slide, there is a navigation bar with icons for back, forward, and search, and the text "Pawan Goyal (IIT Kharagpur)" and "Text Summarization - LexRank".

So, what is text summarization? So text summarization as the name would say that you have a lot of information with you and you want to produce a summarized form. So, you want to summarize this information to some reasonable extent. So, you are trying to distill information and you are trying to present only a summary out of that. So, there are some definitions that you can find for this task of text summarization.

For example, so then Hovy in 2008 gave this definition, so a summary is a text that is produced from one or more texts that contain significant portion of the information in the original text and is no longer than half the original text, so what do you see here?S

So, you have some initial data text data; it can be some documents one document set of documents or whatever and that contains some information, you want to convert that into some different form. So, that will be again text but now it is much more smaller than the original data. So, it can be may be half or even one-fourth of the original data; what is important is that, it should contain the most the important information from the original text. So, you want to retrieve the original important information from there and try to compress otherwise that is the goal of summarization.

So, we can also give this definition that it is the process of distilling the most important information from a source to produce an abridged version for a particular user or task. So, it might happen that you need a very generic summary or it might be for a particular user or a task, so that is also possible in the case of summarization.

So, now so when we talk about summarization, so you might have written summaries yourself when in your school days and all and it is said that humans are generally very good in doing summarization. So, they can not only reduce it to half, one-fourth they can also go down to the very critical bit what is the actual information that is there. So, this is like a; so Calvin Coolidge, so he heard a lecture for 4 hours on a clergyman preaching on sin and somebody then asked what did he say; so he can mention that in one bit that he; so that person said that he is against it. So, this is like condensing the information to the very critical bit, so humans are very very good at that.

But what about the machines, so what can machines do. So, what how can they achieve a summary from a given text.

(Refer Slide Time: 03:42)

**Automatic Text Summarization**

**Goal of a Text Summarization System**  
To give an overview of the original document in a shorter period of time.

**Summarization Applications**

- outlines or abstracts of any document, news article etc.
- summaries of email threads
- action items from a meeting
- simplifying text by compressing sentences

Pawan Goyal (IIT Kharagpur)      Text Summarization - LexRank      Week 11, Lecture 1

So, when we talk about automatic text summarization, so we are talking about an algorithm or a system that will take an input as a document or set of documents and for a task, it will produce a summary out of that. So, the goal would be to give an overview of the original document in a shorter period of time. So, we can see this definition with these different applications of summarization. So, for example, you are seeing suppose you want to read certain research article or a news article. So, it would be good if somebody can present a short summary of that user research article and then you can make your mind whether you want to read this whole article or not.

So, in scientific articles you get the abstracts as the summary, so abstracts tell us about what do you see in this research article, but news articles you have only headlines, you may not have the abstracts. Suppose you take news article and produce a summary of the what is the main information that is contained, that can also help you in saving time in if you want to read a lot of different articles or it can also help you understand whether you want to read it in more details.

Then you can talk about summaries of email threads, there are email threads between it may be between customers and different agents for a particular product, it can be about a particular research task that you are having lot of conversations you want to find out what are the important points from this big email thread. We can talk about summarizing meeting minutes, section items for a meeting or you have a lot of sentences and you try to combine them and

compress information from them. So, there can be many many different applications here that you can think of for text summarization.

(Refer Slide Time: 05:39)

The screenshot shows a search results page with a blue header bar containing the title 'Application: Generating Snippets'. Below the header, the search query 'what is the relation between pressure and velocity' is entered. The results are categorized by source: Web, Images, Videos, News, More, and Search tools. A snippet from a news article about Robert O'Neill is displayed, followed by another snippet from a news article about Ambati Rayudu. The search results also include snippets from physics forums and books, such as 'fluid dynamics - Relation between pressure, velocity and ...' and 'Bernoulli's Equation'. At the bottom of the page, there is a photo of a man (Piwan Goyal) and text indicating the slide is 'Text Summarization - LexRank' and 'Week 11, Lec 1'.

Here is one application that you see like daily that is generating snippets from the web pages. So, you are searching for something you get some results, but with the results you also get some snippets; it is like here the news Robert Oniell taking credit for killing Osama Bin Laden sparks debate. So, this is from old news article and with that you also see a snippet from the news. Similarly if you do some search what is the relation between pressure and velocity, you get different results and with some results you can also get the snippet that is relevant to the question that you have asked and contains the important information from the document. So, this is another very important application for text summarization.

(Refer Slide Time: 06:25)

*Automatic Text Summarization*

*Genres of Summary*

- Extract vs. Abstract
  - ...lists fragments of text vs. re-phrases content coherently.
- Single document vs. Multi-document
  - ...based on one text vs. fuses together many texts.
- Generic vs. Query-focused
  - ...provides author's view vs. reflects user's interest.

Piwan Goyal (IIT Kharagpur)      Text Summarization - LexRank      Week 11, Lecture 1

Now, when we talk about summary there are different genres in which you can produce summary. So, for example you can talk about building an extractive summary or an abstractive summary, so let us try to understand the difference.

In extractive summary, you will take the input document that and you will break it into say various sentences and you will try to pick up what are the important sentences from here, so you are extracting information from there. So, you are extracting some segments from there, but in abstractive summary you will also take information and rewrite in your own words and you will merge different sentences fuse them together and so on. So, in extractive summary you are listing fragments of text; on the other hand in abstractive summary you are taking content and rephrasing that so that becomes much more readable. So, when you are writing a summary you are probably doing it in abstractive manner.

Then you can have a single document summary or a multi document summary. So, you have a single news articles or you take multiple articles that are belonging to the same theme and you are producing the summary out of that, so both are possible. So, this is based on one text and this is fusing different text together, it can be generic summary from the input document. So, what is the generic view of the document or it can be a query focus summary that is you have got the query and you want to get a summary that is focused on your particular query.

So, the document might contain about contain many different informations but the information that is close to your query that will be summarized. So, you can also think it as a

complex question answering task, you are asking a question the information is there in multiple documents, you find out information and reply to you based on what is relevant to your query.

(Refer Slide Time: 08:14)

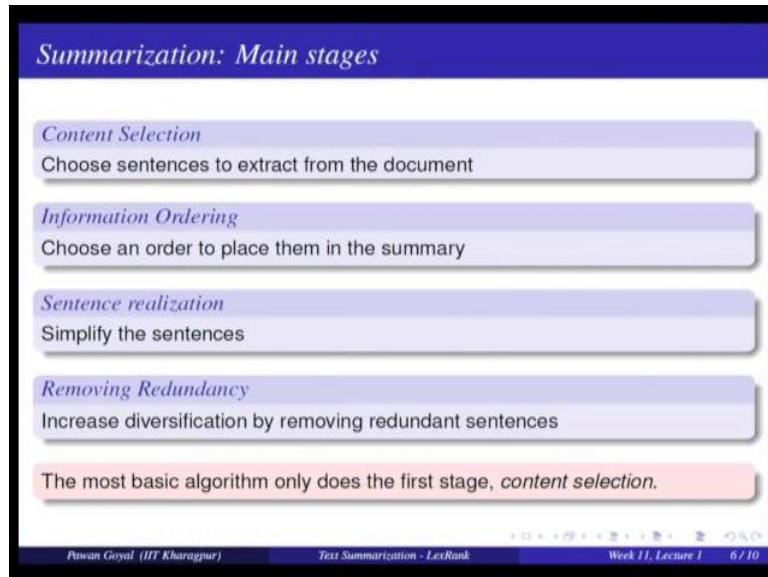
The slide has a blue header bar with the title "Automatic Text Summarization". Below it is a purple box containing the heading "Genres of Summary" and three bullet points:

- Extract vs. Abstract
  - ...lists fragments of text vs. re-phrases content coherently.
- Single document vs. Multi-document
  - ...based on one text vs. fuses together many texts.
- Generic vs. Query-focused
  - ...provides author's view vs. reflects user's interest.

Below this is a pink box with the text: "Query-focused summarization can be thought of as a complex question answering system". At the bottom right is a circular profile picture of a man with glasses and a blue shirt. The footer contains the text "Piyan Goyal (IIT Kharagpur)", "Text Summarization - LexRank", and "Week 11, Lecture 1".

So, in our next two three lectures we will mainly focus on extractive single document and generic summarization. So, we will talk about how do you take a document and create an extractive and generic summary out of that, but whatever method we will discuss are very easily extendible to the other forms like multi document summary and query focus summary and also for abstractive summary, we will give some discussion that how do you use this method for generating an abstractive summary. So, let us focus on the simplest task that can help us to talk about only the fundamentals that what are the different algorithms or hypothesis that you will use for constructing a summary from a given input document.

(Refer Slide Time: 09:06)



So when we talk about summarization what are the main stages that one need to go through so. Firstly, you are given an input document first task would be you will try to select the content that is important for summarization; that is find out what are the most informative sentences from here. So, if I am talking about extractive summarization; find out important sentences from this document. So, you will see how do we design an approach for finding the important sentences, what are the different hypothesis I can take.

Now, once I have selected the content; I know these sentences are important, what would be the next task? Next would be I will try to order them somehow. So, whether this sentence will go first or this sentence will go after this; this also depends on how will I make it more readable or more presentable to the end user; there I have to talk about how it can be more coherent and so on.

Then I might want to simplify the sentences that I am getting from the actual summary; that is I want to remove some of the redundant phrases and so on; this is like simplification. Once I have done that, I may also want to remove the redundancy that is I found what are the important sentences but there two of the sentences are very similar to each other. So, I want to remove a sentence that is similar to some of the sentence that is already present in my summary. So, I want do not want to give the same information multiple times because I have a shortage of space.

So, whenever I talk about summarization; I generally put a restriction that I want for a document I want a summary of length 100 or length 200. So, I do not want to waste the space by giving the same information multiple times, this is like getting also diversity in the information. So when we talk of the most basic algorithm, this will only concern the first part that is the content selection; how do I find the most important sentence in the document, so let us see what is the simplest algorithm.

(Refer Slide Time: 11:17)

*Unsupervised content selection; Luhn (1958)*

*Intuition*  
Choose sentences that have salient or informative words

*Two approaches to define salient words*

- *tf-idf*: weigh each word  $w_i$  in document  $j$  by tf-idf

$$weight(w_i) = tf_{ij} \times idf_i$$

- *Topic signatures*: choose a smaller set of salient words, specific to that domain

$$weight(w_i) = 1 \text{ if } w_i \text{ is a specific term (use mutual information)}$$

*Weighing a sentence*

$$weight(s) = \frac{1}{|S|} \sum_{w \in S} weight(w)$$

Pawan Goyal (IIT Kharagpur)      Text Summarization - LexRank      Week 11, Lecture 1      7 / 10

$$weight(w_i) = tf_{ij} \times idf_i$$

$$weight(s) = (1/|s|) \sum_{w \in s} weight(w)$$

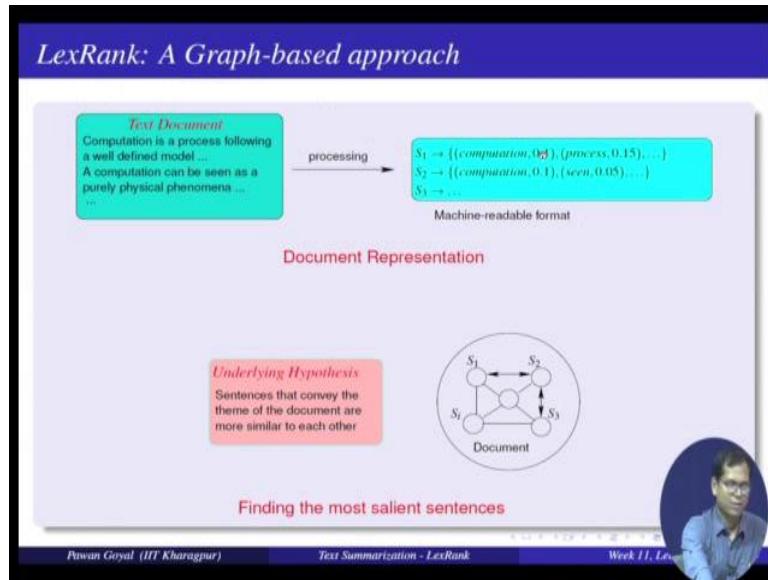
So, this was designed by Luhn in 1958; this is called unsupervised content selection, what is the idea few sentences that are very very informative and that how do you realize the sentences that are very informative. So, find the sentences that are having very salient or informative words; sentences that are having very informative words choose those. So, immediately the question is; for doing that I need to define what is the information contained for each of the word and given sentence I can add the information for each of the words.

So, how do I assign the information for each of the words, so what are the simple measures. So, one simple measure is you take a tf-idf for each word, so for each word in the document in the sentence, you find that what is the tf-idf of it. We already discussed how do we compute tf-idf of it and then take an average tf-idf for the whole sentence. Second approach would be, you define what are your topical words, what are your content this domain specific words and then take sentences that contain more of these topic bearing words. So, that is you define something like a topic signature and generally choose a smaller set of words that are specific to that domain and weight of word  $w_i$  can be 1, if  $w_i$  is a specific term in that domain and 0 otherwise. So, that is you are talking about politics, you know there are certain terms that are very important these terms into politics and they are called the topic signature for this domain and whichever word is in this topic signature is given a weight 1 otherwise 0.

So, now once you have defined a weight to each of your word using one of these methods, you find the weight of your sentence by taking adding the weight of each and every individual word and dividing by the length of the sentence; that is what is the average information of the words in the sentence, so you take the information for each of the word and take an average. This is a very very simple method, it does not see what the other sentences in the document and an anything else; it just assigns the information to each of the word independently and takes the important sentence by this method.

So, this is the simple method that you can use, but what is more much more principled method and much more common method is something called a LexRank method, this is coming from a this is like a graph based method for summarization.

(Refer Slide Time: 13:45)



So, what is that approach, so this flowchart kind of explains this whole approach; let us try to understand this. So, I have an input text document that contains a set of doc; set of sentences. So, like here you have a sentence like computation is a process following a well defined model and so on and second sentence a computation can be seen as a purely physical phenomena and so on so you have a set of sentences here.

So what is the first step, you find out what are the individual sentences here and provide their tf-idf weight. So, you say sentence S 1 contains the word computation with a tf-idf weight of 0.1, process with a tf-idf of 0.15 and so on; sentence S 2 contains computation with weight of 0.1 seen at the weight of 0.5 and so on and I will do it for all the sentences. This is like I am converting this document to a machine readable format, where I have different sentences; each sentence has some words with their different tf-idf weights.

Now once I have done that, what is the next step? Now I treat it as a graph, where all these sentences are the nodes in the graph and edges depend on what is the cosine similarity of different sentences in this the pair of sentences. So, what I will do now; I will take all the sentences in my document and I will construct a graph where these are the nodes and the edges are what is similarity between every pair of sentence S 1 and S 2; how similar they are.

So, this graph is again very easily constructed because you already know what is the representation for each sentence; so you can come to the cosine similarity to get the similarity between two sentences. Now, once you have this graph then you use an hypothesis for getting

the important sentences and what is the hypothesis that is the most important part here. So, the hypothesis says that when you are talking about the document, so it is a news document or research article; it is about the theme right; you have an underlying theme there.

Now sentence that are important to the document should be talking about that theme and because this theme is prevalent, this sentence should be similar to many other sentences in the document. So, that is a sentence that is similar to many other sentences should be called as the important or the relevant sentence and this is the main hypothesis that is used here and these other sentences should also be relevant. So, sentences that convey the theme of the document are more similar to each other.

So, now this can give us a nice intuition on how do we construct an algorithm; sentences that are talking about theme of a document are similar to each other. So, I am saying a document is important or a sentence is important; if many other important sentences are similar to it, it is similar to many other important sentences and that if you try to think of this hypothesis that is very close to what we do in the case of page rank algorithm. So, in page rank we say that a web page is important, if many important web pages link to that and here we are saying a sentence is important; if many important sentences are similar to that. That means, similarity we have to somewhat link it to incoming edges and so we directly convert that to some sort of a page rank algorithm, so what do we do there?

(Refer Slide Time: 17:19)

**Sentence Centrality Measure**

*Finding the most salient sentences*

PageRank based algorithm is used to compute the sentence centrality vector  $I$ .

$$\tilde{M} = \begin{bmatrix} 0.0 & 0.5 & 0.0 & 0.4 & 0.1 \\ 0.5 & 0.0 & 0.5 & 0.0 & 0.0 \\ 0.0 & 0.5 & 0.0 & 0.5 & 0.0 \\ 0.4 & 0.0 & 0.4 & 0.0 & 0.2 \\ 0.3 & 0.0 & 0.0 & 0.7 & 0.0 \end{bmatrix}$$

$$I_j = \mu \cdot \sum_{\forall k \neq j} I_k \cdot \tilde{M}_{k,j} + \frac{1 - \mu}{|S|}$$

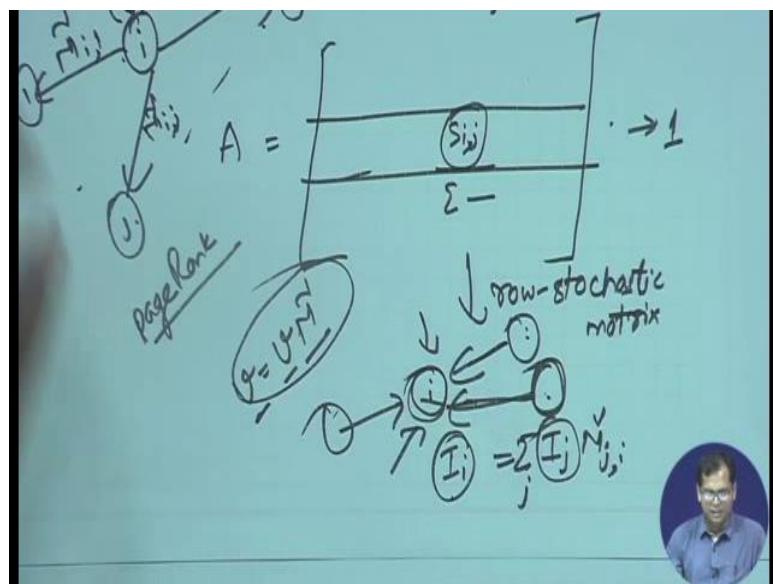
$$I = [ \begin{array}{ccccc} 0.22 & 0.18 & 0.2 & 0.3 & 0.1 \end{array} ]$$

Pawan Goyal (IIT Kharagpur)      Text Summarization - LexRank      Week 11, Lecture 1

So, we construct a document graph with sentences as the vertices that is easy I know how many sentences are there that many vertices I construct. Then I compute the similarity between sentences and I construct some sort of a matrix. So, I compute similarity between S 1 and S 2, S 3 and S 4 and so on by cosine similarity and then I have a matrix format; of all the n cross, n pairs if there are n sentences in my document, I have a matrix of size n cross n that stores how similar each two sentences are.

Now, once I have constructed this matrix, I want to apply a page rank kind of algorithm to compute the informative score for each node. So, that is I want to compute the sentence centrality vector  $I$  and  $I$  has as many elements as the number of nodes. So, I will have a score for each of the node and the node that gets the highest informative score, has the highest page rank or is the one that is selected first. So, let me try to spend some time in discussing what is the rational for using this approach and how do you actually apply this algorithm.

(Refer Slide Time: 18:46)



So, what you are doing here, so you are constructing this matrix  $A$ ; that is how similar two sentences  $i$   $j$ . So, the element  $i$   $j$  will tell me how similar two sentences  $i$   $j$  are, so this is an initial matrix. So, now to be able to apply page rank algorithm; we need to convert that into a row stochastic matrix. So, convert that to a row stochastic matrix; what do I mean by row stochastic matrix; such that all the elements in an individual row add up to 1 and how do you do that? You will take all the elements sum them together; get a addition and divide it by the

summation over all these elements and this will ensure that all the elements add up to 1, so summation of all these elements will add up to 1.

Now how does that help? So there we need to understand what is the intention for and this is called the  $M \tilde{}$  here, this is my  $M \tilde{}$  that is the row-stochastic matrix. Now what is the intention of adding this to 1, so adding to 1 gives me some sort of feel of probability right.

So, how we can see these integers probabilities, so now this  $M \tilde{i} j$  will be the probability with which you are going from node  $i$  to node  $j$ . So, that is assume you are having a graph node  $i$  and there are all other nodes node 1, node 2, node  $j$  and node  $n$  up to node  $n$ . So, idea is that you are doing a random walk on this graph; at some point of time you are at node  $i$ ; from node  $i$  you can go to all these nodes. So, there is a probability with which you can go to different nodes that is  $M \tilde{i} 1; M \tilde{i} 2, M \tilde{i} j$ ; there can be some probability of staying here  $M \tilde{i} i$  also; there is some probability with which you can go to different nodes and so you have distribution and you will sample some node and you will go to that and suppose you go to node  $j$ , then you start from there. So, this is a random walk that you do and when you do this random walk, it converges to something like  $v$  is equal to  $v M \tilde{}$ , so  $v$  is the actual page rank vector that you are obtaining from there.

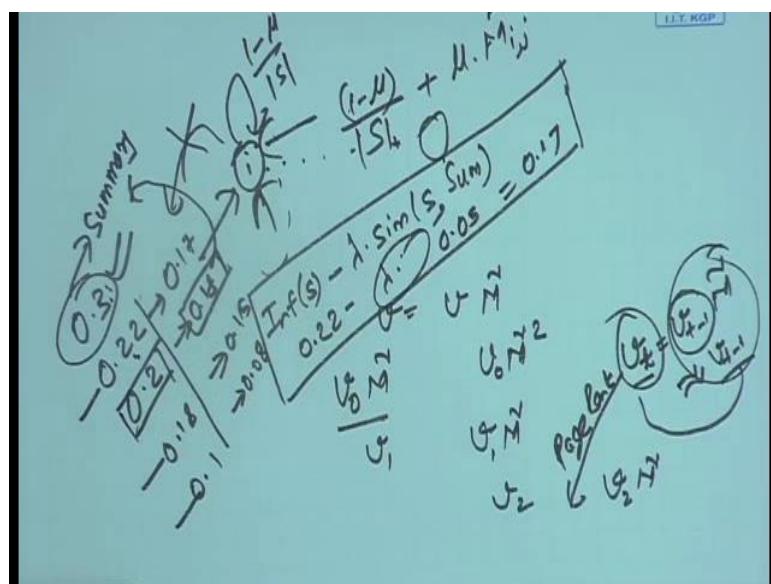
So, here in page rank what is important; a node, a webpage is important if many important web pages link to that. So, that is I have a node  $i$ ; it is important if it is getting links from many other important pages. So, this is important, this is important, this is important and it is getting inlinks to many of these pages and how do they correlate. So, because you are getting inlinks from many important pages what does that mean?

So, the translation between this random walk and this page rank value is the high page rank value indicates that, if you do a random walk for sufficiently long number of time; you will stay on that node for a larger fraction of time. So, suppose this node has a higher page rank than this; that means, if you do a random walk for large number of time; this node, you will stay at this node for longer time than at this node. So, now if this node is linking to this node; lot of lot many times what does that mean because you will be staying at this node for longer times and from here you have a good probability of reaching to this node. So, accordingly you will also stay at this node for a longer time and if you have it from many important nodes; that means, there is a high probability that you will be staying at this node for a longer number of times.

So, that is if a node gets incoming edges from many important nodes then it has a high page rank and that is formulated by this simple equation. So, that is if I want to write it as a equation; I will say importance of this node  $i$  would be importance of node  $i$  will be if I write crudely I will say will be proportional to importance of all the other nodes  $j$ , that are linking to it and the way the probability with which they are linking to it times  $M \tilde{M}_j i$ . So,  $i j$  times  $M \tilde{M}_j i$  and I will sum over all  $j$  and you will see that this will actually take it to this formula  $v$  is equal to  $v M \tilde{M}$  and that is my page rank algorithm; importance of a node is; importance of all other nodes that are giving incoming link times the probability of going from that node to this node; how high this incoming linkage.

So, and then you iteratively compute these  $i$ ; all these  $i$  values and there is a very simple formula for computing these iteratively and what is that.

(Refer Slide Time: 24:26)



So, we know the equation is  $v$  is equal to  $v M \tilde{M}$ , so how do you compute it; you start with some initial  $v_0$  and keep on multiplying with  $m \tilde{M}$ . So,  $v_0 M \tilde{M}$  is your  $v_1$  then you got  $v_1 m \tilde{M}$ , so this will be like  $v_0 M \tilde{M}^2$  then you got this is your  $v_2$ , then you go  $v_2 M \tilde{M}$ , keep on doing that. Now what will happen because this is my equation, it will converge at some time where  $v_t$  that is  $v_{t-1} M \tilde{M}$  is very similar to  $v_t$  so; that means, now  $v_t$  and  $v_{t-1}$  are roughly same and this is your actual equation.

So, it will converge at some time, so you keep on multiplying by  $M \tilde{M}$  at some point of time and at every time you keep on finding the difference;  $v_t$  minus  $v_{t-1}$  find out the

difference; at some point of time this will be below threshold at that time you will stop and whatever  $v_t$  minus 1 or  $v_t$  you get at this point; this becomes your page rank and that is how you compute your page rank.

So, now we saw what is the rationale of using page rank and how do we compute page rank, but now if you go to the actual formula; this looks slightly more complicated than what we were saying. Now this is similar right  $i_j$  is equal to summation over  $i_k$  times  $M_{\tilde{k}j}$  importance of  $k$  and the probability of going from  $k$  to  $j$  summation over  $i_k$ , but you see something like a  $\mu$  and  $1 - \mu$  divided by  $S$ . So, let me see what is this  $\mu$  and  $S$ ;  $S$  is the number of sentences here and  $\mu$  is something like a small teleport probability what is the idea?

Suppose in your whole graph; there is a node  $i$  that has some incoming edges, but there is no outgoing edge from here, so there is no outgoing edge. So, when you are doing a random walk; once you come to node  $i$ , you cannot go back; you cannot go to any other node in the graph and your random walk is stuck and this creates a problem in defining the statistic probabilities and all that. So, that is why you say from each node I will assign a small probability to all the nodes.

So, this probability is nothing, but  $1 - \mu$ , we equally distribute among all the  $S$  nodes divided by  $S$ . So, there is a probability  $1 - \mu$  divided by  $S$  going to all the nodes, they will always have self probability  $1 - \mu$  divided by  $S$  and to balance the probabilities; you multiply plus  $\mu$  and the actual probability  $M_{\tilde{i}j}$ . Suppose this was  $M_{\tilde{i}j}$  initially, now the probability becomes  $1 - \mu$  divided by  $S$  plus  $\mu M_{\tilde{i}j}$  and you will see that now the probability will still add up to 1 and that is the only thing that you are seeing here, so  $i_j$  is  $\mu$  times this plus  $1 - \mu$  divide by length of the (Refer Time: 27:55).

So, now if you take in general the value of  $\mu$  that you will take here is roughly close to 0.85 that is a common general value. So, if we take  $\mu$  is equal to 0.85 and try to run this algorithm, so I will encourage that you try to run this algorithm, you will find that these are the page rank values that you will get for these five sentences. Now how would you pick a sentence from here, you will take the one that is having the highest score. So, we will take the sentence  $S_4$  first, then if you want a longer summary then you will take  $S_1$  further longer

then S 3; more than that will be it will not be a summary. So, we will take S 4 then S 1 and S 3 if required, so this is how you will construct a summary.

That is the first part of content selection, how do we select the important sentence from the document. Now there will be one problem here that is suppose I select the S 4 sentence, but S 1 is quite similar to this, but this is coming how to be irrelevant as per this approach. So, I want to give some less weight to a sentence if it is similar to some already existing sentence in the summary and that is why I can do some sort of diversity and I will quickly tell what will be the approach we will use for diversity and this is called the maximum marginal relevance.

(Refer Slide Time: 29:19)

*Removing Redundant Sentences*

*Maximal Marginal Relevance*

- An iterative method for content selection from a selected list of important sentences
- Iteratively choose the best sentence to insert in the summary that is minimally redundant with the summary so far ( $Sum$ )

$$Inf(s)_{MMR} = \max_{s \in D} (Inf(s) - \lambda \cdot sim(s, Sum))$$

where  $Inf(s)$  denotes the informativeness score of a sentence

Pawan Goyal (IIT Kharagpur)      Text Summarization - LexRank      Week 11, Lecture 1

$$Inf(s)_{MMR} = \max_{s \in D} (Inf(s) - \lambda \cdot sim(s, Sum))$$

So, given something is already in the summary; how much more relevance you are bringing in by the new sentence. So, this is like an iterative method for content selection from a selected list of important sentences. So, what is the idea; you iteratively choose the best sentence to insert in the summary that is minimally redundant with the summary so far. So, informativeness of the sentence will be among all the sentences in the document; informativeness of the sentence minus lambda times similarity of sentence with the already

existing summary and whichever gives the highest informativeness is included in the summary, so what is the idea?

So, suppose you have the sentences with this informativeness 0.3, 0.22, 0.2, 0.18 and say 0.1 informativeness of five different sentences. So, you took this sentence for your summary; summary contains this sentence one. Now as per the relevance, you will take sentence S 2, but what this approach says; now you again re rank them, what is the criteria; this would be informative score of a sentence S minus lambda times similarity of S with the summary. So, that will be 0.22 minus lambda times similarity of this sentence with this sentence; suppose it comes out to be lambda times similarity this whole thing comes out to be 0.05, so this becomes 0.17. You will do that for all the sentences, suppose it comes out to be 0.17; this comes out to be 0.18, this comes out to be 0.15, this is 0.08. So, then you will again sort them and take the one with the highest value as per MMR and this goes to your summary.

Now your summary has two sentences; again you will compute this informativeness minus lambda times similarity of this sentence with all the sentences in your summary. You will again rank this, this and this and you will try to take the one that is having the highest score as per MMR and you will do that iteratively until you have got the desired length of your summary and that is the idea of taking redundancy.

So, you can see that how we are helping, so initially I would have taken this sentence followed by this sentence, but this sentence is very similar to this sentence. So, this approach helped me to choose another sentence that is not so close to the original sentence, so you are also having the diversity, so this was one approach for doing summarization. So, in the next lecture we will talk about another approach for doing summarization that is based on some sort of optimization. So, see you then.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 53**  
**Optimization Based Approaches for Summarization**

Welcome back for the lecture 2 of this week. So we had started our discussions on text summarization, and in the last lecture we discussed one of the very important approaches for text summarization that is based on a graph based method, that is called also lexrank or page rank method for summarization, what was the idea there? So we were taking all the different sentences in the document and then finding out the similarity between them and trying to choose and then once we run the page rank algorithm over this graph, we take the sentences that have the high page rank.

So idea is that my document corresponds to one particular theme and the sentences that are important to this document should have many other sentences similar to that. And that was the idea that was captured by using the page rank formulation. So the sentences should have high incoming edges from other important sentences.

And then we said this will give you the important sentences, but further if you want to find out sentences that are diverse enough with respect to each other then you will use some sort of mmr maximum marginal relevance algorithm. So you will say that sentence if it is already similar to some sentence in the summary I will reduce its overall score. And then I will sort the remaining sentences again and choose the sentences that are coming out to be having highest score. So lexrank you can always choose as one of the very simple and effective base length for any of the summarization tasks. Although that it does not give you much flexibility in having you decide what is your criteria for importance. So how do you want to choose your important sentences?

So in this lecture today what we will see that there are some approaches that that can help you that can give you a generic framework for optimization. And then it is up to you to define your own optimization function your own constraints and you can keep on adding your constraints depending on your task. So you will probably discuss about 1 or 2 scenarios that how do you modify it for different tasks. So in this lecture today. So we are talking about the optimization based approaches for summarization.

(Refer Slide Time: 02:30)

*Global Inference*

- Let us define document  $D$  with  $t_n$  textual units

$$D = t_1, t_2, \dots, t_{n-1}, t_n$$

- Let  $Rel(i)$  be the relevance of  $t_i$  to be in the summary
- Let  $Red(i,j)$  be the redundancy between  $t_i$  and  $t_j$
- Let  $l(i)$  be the length of  $t_i$

Pawan Goyal (IIT Kharagpur)   Optimization Based Approaches for Summarization   Week 11, Lecture 2   2 / 8

So for doing that, let us define it formally define a document as containing some different textual units and because we are talking about extractive summarization. So our textual units will be sentences. So I have some  $n$  number of sentences. So I am denoting them by  $t_1$  to  $t_n$ . Now let us say I can define what is the

$$D = t_1, t_2, \dots, t_{n-1}, t_n$$

relevance of a textual unit  $t_i$  to be in the summary by relevance of  $i$  rel  $i$ . Now this is a generic formulation. So I am saying I will define the relevance of each unit by  $rel(i)$ , but how do I compute  $rel(i)$ , can you are free to choose your own method.

So what are the different methods of computing relevance that we have discussed? One is simply by taking the weights or topic signatures in the sentence and adding or and doing the average over all the weights in the sentence. That is one particular method of choosing the relevance of  $i$ th textual unit. Then you can also run your lexrank algorithm and find out the relevance as the page rank value of that sentence, but suppose you want to use some different criteria for relevance. So you are free to do that. So your relevance criteria might be how many named entities are involved in this sentence. That can be your criteria then you can accordingly choose your define your relevance.

So you can define your relevance then you have to also define what is the redundancy between  $n$  into textual units. So that is how much they are redundant to each other. And why we are doing that? Because in the final summary that we obtain we do not want multiple sentences that are conveying the same information, remember in summarization

the idea is that I want to get as much information as possible within the minimum possible space. So I always have a space crunch. So I want to fit as many information as possible. So I want to remove redundant sentences. So that is another function that you have to obtain.

Now the redundancy between 2 textual units again you can define it by simply taking the cosine similarity of them, how similar they are or you can also define it by some other methods like how much apart they are in the document are they very near are they very far apart. You can also depend on many different criteria like do they talk about the same sort of entities and many other things. So once you have done that, and let us say we define what is the length of the textual unit that is simply the number of words that are there in that textual unit.

Now, once we have all these definitions what will be my optimization criteria? So for choosing the summary, when I talk of summary, so in all these systems I will generally have an upper limit on the summary. So I will say I want a summary of length  $k$  and  $k$  will be roughly 100 or 200 at most depending on your input document. So you want a summary of length  $k$ . So what should be some optimization criteria?

Now, optimization criteria should be from all these  $n$  units select a subset such that the score of the summary is maximized. And how do you define the score of the summary? The summary will have some set of textual units. So score could be summation over the relevance scores of all these units that you have taken in the summary minus what is the redundancy between any 2 units. So that will be all paired redundancy score. So relevancy score of all the units in the summary minus redundancy score of all the pairs in the summary, that will be your optimization function and you want to select the subset such that it maximizes this score.

(Refer Slide Time: 06:16)

Inference Problem

- The inference problem is to select a subset  $S$  of textual units from  $D$  such that summary score of  $S$ , i.e.,  $s(S)$ , is maximized.
- $S = \arg \max_{S \subseteq D} \left[ \sum_{t_i \in S} \text{Rel}(i) - \sum_{t_i, t_j \in S, i < j} \text{Red}(i, j) \right]$   
such that  $\sum_{t_i \in S} l(i) \leq K$ , where  $k$  denotes the maximum length of the summary

Pawan Goyal (IIT Kharagpur)      Optimization Based Approaches for Summarization      Week 11, Lecture 2      3 / 8

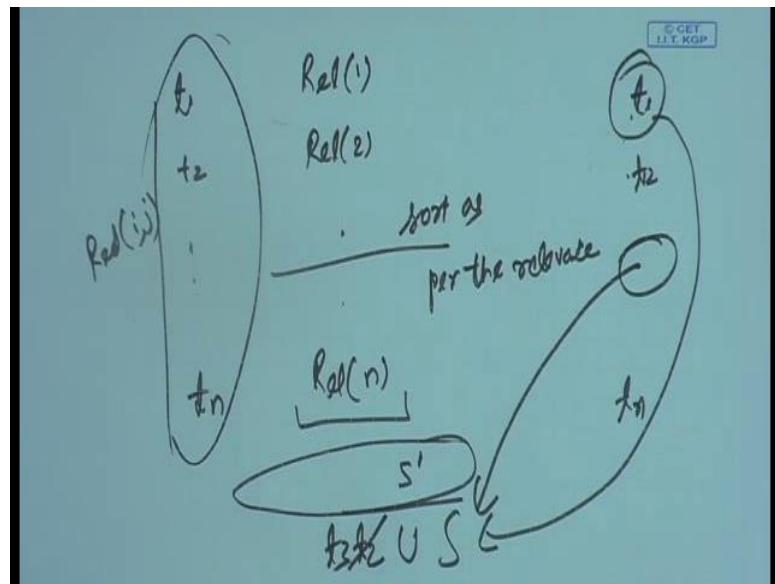
] So I can say that what is my

$$S = \arg \max_{S \subseteq D} \left[ \sum_{t_i \in S} \text{Rel}(i) - \sum_{t_i, t_j \in S, i < j} \text{Red}(i, j) \right]$$

inference problem. I want to select a subset as of textual units from d. So d is my whole document containing t 1 to t n. Such that summary score of s we can define it as small s of the set capital S is maximized. And how do I define the summary score of the subset, as I said I will take all the textual units there get their relevancy scores sum over that, summation over relevance of i for all the units  $t_i$  in  $s$  minus summation over redundancy score of any 2 units, i and j that I am taking in the summary and that I will sum over all the pairs, but I do not want to sum over I do not want to repeat a pair. So I will put some ordering say I have to be less than j, and they can be in the order in which they occur in the document. So I have this  $t_i, t_j$  in  $s$   $i$  is less than  $j$  and I am getting the redundancy score. So what is the objective function trying to achieve maximum relevance within the set with the minimum redundancy? So total score is defined like that.

Now, and then what is the constraint? For doing that, the length of all the textual units that you are doing the summary  $l_i$  in all the  $t_i$  in  $s$  should be less than equal to  $k$ , that is your constraint and so capital  $k$  denotes the maximum length of the summary. So you have a constraint and you have an optimization function. Now question is that how do you achieve this optimization, how do you maximize this particular objective function. Now in the case of MMR we were using a sort of greedy approach for doing that. So a greedy approach is always a solution. So how would you use a greedy approach?

(Refer Slide Time: 08:13)



So you have some  $n$  textual units and you have also completed the relevancy scores say units  $t_1, t_2$  up to  $t_n$  and you have completed the relevancy score  $rel_1, rel_2$  up to  $rel_n$ . And as we have discussed this can be completed by many different manner ways. So let us not worry about that.

Now, once you have completed all that you want to maximize your objective function that is and yeah and yeah you have also completed your redundancy score of  $i j$  between every 2 units. So your objective function is I want to; somebody said that summation of relevance is minus redundancy between all the pairs is highest. So what you will do in greedy approach? You will first sort these, sort as per the relevance and you get some final sorting order. And let us say that order is  $t_1$  to  $t_n$  –  $t_1$  has the highest relevance and  $t_n$  has the lowest relevance. So what would be the greedy approach? I will say take  $t_1$  in my summary already. So now,  $t_1$  is there in the summary because it has the highest relevance.

Now the next unit will not be directly  $t_2$ . So we will say among all the possible units which unit I can add to my summary such that the score is maximized. So I am doing it one by one. So what is the score? Summation over relevance of  $t_1$  plus  $t_2$  minus redundancy of  $t_1 t_2$ , so that will the score for adding  $t_2$ , you will add  $t_2$  to the summary. So this is your summary you will add  $t_2$ . And now this you call it your  $s'$

prime and then compute your objective function for s prime with t 2. Then you will instead of t 2 you will put t 3 and compute it and so on you will do for all the t n.

And finally, you will have all different summaries all the scores take the one that is having the maximum score. So suppose some t 3 gets the maximum score. So this gets into my summary. Now again from the remaining units again keep on adding one by one to the at a time to the summary and see which one is giving you the highest score. Take the one and you will stop whenever you we have achieved the desired length of your summary that will be your final summary. So it is a greedy method.

(Refer Slide Time: 10:41)

### A Greedy Solution

```

1.   Sort D so that  $Rel(i) > Rel(i+1) \forall i$ 
2.    $S = \{t_1\}$ 
3.   while  $\sum_{i \in S} l(i) < K$ 
4.        $t_j = \arg \max_{t_j \in D - S} s(S \cup \{t_j\})$ 
5.        $S = S \cup \{t_j\}$ 
6.   return S

```

So what we are doing? So I am sorting my all the textual units my document such that relevance of ith unit is getting the relevance of i plus oneth unit for all t i. So you will get the score t 1 to t n, they are in the sorted order. Then what do you? Do you take the t the first unit in your summary? Then while you have not achieved the real length of the summary, for this loop you will take each sentence that is remaining each unit that is remaining put that in your summary as union t j and compute the score and get the r max over all t j's. And that will be the one that you will include in your summary whichever gets the highest score. And then you keep on doing that until you have achieved the desired length of your summary.

And this is a very generic approach you have to define your own relevance score redundancy score and then you have to apply this algorithm to obtain a summary such

that it has the highest relevance, but minimum redundancy. So you achieve both relevance and diversity at the same time by using this approach.

Now, we are saying that this is a greedy approach. Why is this greedy approach? You are picking you are not going to an optimal solution you taking the greedy approach that say it is near optimal solution it may not be an optimal solution. I am always choosing  $t_1$ , but it might happen that in my optimal solution I do not have  $t_1$  because  $t_1$  is similar to many other sentences. This can always happen. So this does not guarantee in optimal solution, but this is good enough if you want a decent solution. There are other approaches where you can use dynamic programming algorithm to find out what is the best way I can fill up my length  $k$  such that this is maximized of course, if you try out all possibilities, it might be exponential in nature, but you can always in dynamic programming. So that you are storing somewhere else what is the best way of achieving the summary of length 5, 10, 15 and so on and then use that for any higher length. So that is another approach.

So what we will discuss instead is an approach that is again very generic. And there although we have started with some optimization function some constraint you are free to choose your own optimization function and also your constraint and still this will give you an optimal solution. And suppose is called integer linear programming. So there you are trying to optimize some of objective function with respect to certain constraints and you are getting some integer solutions for your different parameters such that they maximize the objective function.

(Refer Slide Time: 13:25)

The slide has a dark blue header bar with the text "Integer Linear Programming (ILP)" in white. Below the header is a large white area containing a bulleted list of five items. At the bottom right is a circular video player showing a man in a light blue shirt. The video player includes standard controls like play, pause, and volume. At the very bottom of the slide, there is a thin dark footer bar with three small pieces of text: "Pawan Goyal (IIT Kharagpur)", "Optimization Based Approaches for Summarization", and "Week 11, L1".

- Greedy algorithm is an approximate solution
- Use exact solution algorithms <sup>with ILP</sup>
- ILP is a constrained optimization problem
- Many solvers on the web
- Define the constraints based on relevance and redundancy for summarization

So let us first see what is that. So we will talk about how to use the integer linear programming for summarization. So this is like a greedy algorithm. So greedy algorithm sorry; greedy algorithm that we discussed in the previous slide it gives me an approximate solution; so ILP will help you an exact solution that is the optimal value for this objective function.

Now ILP is like a constraint satisfaction problem. So you define the various constraints and based on those constraints you are trying to optimize your objective function. So important thing is that all your constraints should be satisfied now your constraints will be such that that are needed for your summarization for example, what is one very simple constraint that you will always put that summation over all the units that you have selected has to be less than or equal to  $k$  where  $k$  is the desired length, that is one constraint that you will always put, plus you might have some other constraints based on your task, we will discuss that once we will give the basic approach.

So one good thing about ILP is that there are many different solvers on the web and you can take any of those. And in the format that they have described you will put your objective function and constraints and they will give you the solution based on that. So initially to understand ILP we can what we can do? We can define our constraints and based on only the relevancy and redundancy. So they are 2 things that we talked about in

the previous slide also. Let us define our constraint objective function based on that and then we will see what will be the form of the ILP.

(Refer Slide Time: 15:04)

*Sentence Level ILP Formulation*

*Optimization Function*

$$\text{maximize } \sum_i \alpha_i \text{Rel}(i) - \sum_{i < j} \alpha_{ij} \text{Red}(i, j)$$

*Constraints*

such that  $\forall i, j$ :

- $\alpha_i, \alpha_{ij} \in \{0, 1\}$
- $\sum_i \alpha_i l(i) \leq K$
- $\alpha_{ij} - \alpha_i \leq 0$
- $\alpha_{ij} - \alpha_j \leq 0$
- $\alpha_i + \alpha_j - \alpha_{ij} \leq 1$



Pawan Goyal (IIT Kharagpur)      Optimization Based Approaches for Summarization      Week 11, Lecture 1

$$\sum_i \alpha_i \text{Rel}(i) - \sum_{i < j} \alpha_{ij} \text{Red}(i, j)$$

So here this is my objective function. So now, what we are doing? So again suppose that we have different textual units t 1 to t n. Now I also have defined the relevance and redundancy. So for now again like in the previous case assume that you know how to compute the relevance and how to compute the redundancy between 2 different units. So once you have defined all that now you are saying I want to maximize some function summation alpha i rel i minus summation i. So this is an ordering to ensure that you are doing it only once alpha i j redundancy of i j.

Now one thing you will notice that we are not in the earlier case we are taking a subset you were saying I will take a subset and for that subset I will maximize. This here we are not taking the subset, but instead what do we have here? We have some different parameters alpha i and alpha i j; now what are these parameters? So alpha i is a simple indicator that tells whether the ith unit is included in the summary or not, so I am saying alpha i summation alpha i rel i; so alpha i will indicate whether the ith unit in the summary or not.

(Refer Slide Time: 16:14)

The image shows handwritten mathematical notes on a whiteboard. At the top left, there is a circled equation  $\sum_i \alpha_i \text{Rel}(i) - \sum_{i,j} \alpha_{ij} \text{Red}(i,j)$ . To its left, another circled equation  $\sum_i \alpha_i = 0$  is written with a note "Opt." above it. Below these, a circled equation  $\alpha_{ij} = 0 \forall i, j$  is shown with a note "0,0" next to it. A horizontal line separates these from the bottom section. Below the line, the equation  $\sum_i \alpha_i \text{Rel}(i) \leq k$  is written. To its right, a circled equation  $\alpha_{ij} - \alpha_i \leq 0$  is shown with a note "0,0" next to it. Below these, two more circled equations are shown:  $\alpha_{ij} = 1 \forall i, j$  and  $\alpha_i = 1 \quad \alpha_j = 0$ .

So alpha  $i$  should have a value of either 0 and 1, 0 or 1. It can take only one of these values. And similarly alpha  $i j$  redundancy of  $i j$  alpha  $i j$  can again take a value only between 0 and only 0 or 1. Now what do these parameters alpha  $i$  and alpha  $i j$  indicate? 0 means it is not in the summary 1 means it is in the summary. So whenever it is 1 you are taking the relevance 0 it is not being counted. Alpha  $i j$  will be one whenever  $i$  and  $j$  both are in the summary and so that; that means, if  $i$  and  $j$  both are in the summary you have reduce you have to subtract the redundancy score. So you are trying to find a solution such that this is maximized. So now, you should put certain constraints.

So for example, if you do not put a constraint I can always find a solution where alpha is equal to 1 for all  $i$  alpha  $i j$  is equal to 0, for all  $i j$  I can always put some solution like that, but you will immediately say this does not make sense because you are not subtracting the redundancy of the sentences, one thing. Second you are taking all the sentences in the summary. So what is the point of the summary you are taking the whole document.

So this optimization function is not sufficient until you add the constraints. So what are the constraints you will add? Since this is the optimization function and what will be the constraints. So constraints the first one that will come to your mind is that the length of the summary has to be less than equal to  $k$ . So how will you put that constraint? You see

what is the; what is by this approach you want find the optimal solution for all this alpha i's in alpha i j's. They can take value between 0 values 0 or 1. What is the optimal values. So you will put some constraints one constraint will be summation alpha i li less than or equal to k, li is the length of the ith textual element. This is less than equal to k. That will be the constraint that will immediately say this kind of solution is not acceptable, but again you can find a solution where some alpha i's are 1, but all the alpha i j's are 0. So you do not want that. You want that whenever alpha i and alpha j are 1 alpha i j has to be 1. So you need some more constraint for that. So what are the constraints we will put here? So we will say. So alpha and also you do not want alpha i j to be one and alpha i to be 0. It cannot happen.

So all these things how do we put together in a constraint let us see. So we are saying all the alpha i alpha j are either 0 or 1 that is the first thing. We are saying and summation i alpha i li is less than equal to k. Then the next 2 constraints are alpha i j minus alpha i is less than equal to 0. So alpha i j minus alpha i less than equal to 0 what is that mean.

They can take values 0 or 1. So what are the possible values? Both can be one this is allowed right alpha i is one alpha i j is 1 is allowed both are 0 that is allowed, but what is not allowed. So and what is and also if alpha i is equal to 1 alpha j is equal to 0 sorry alpha i j is equal to 0. That is also fine this will be minus 1 what is not allowed. Alpha i j is equal to one alpha i is equal to 0, this is not allowed. Because then this will be 1. And why it is not allowed? What is it saying is that you are not including the ith unit, but the redundancy between i and j and this is not possible this you can pick only when alpha i is equal to 1. So this is a constraint like that similarly you can state now alpha i j minus alpha j less than equal to 0 and that you can understand in the same manner.

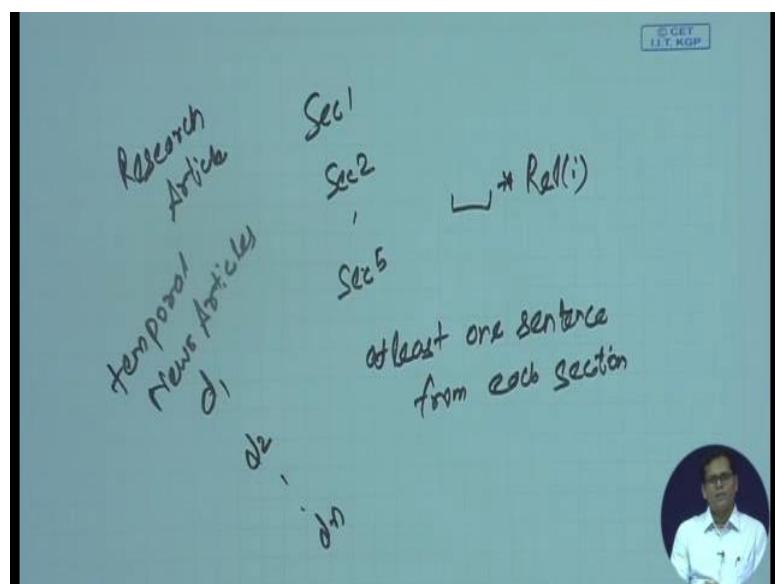
Now, what is the third constraint you are putting? That is alpha i plus alpha j minus alpha i j is less than equal to 1. Now what is this saying? So this can take all the values except a value of 2. Now when will this take a value of 2? So you can see that it cannot take a value of larger than 2 because it can be at most 1 at most 1 it can be at least 0. So maximum value is 2 and it is allowed to take values up to 1.

So what happens when it takes value of 2? That means, it is 1, it is 1, it is 0, but it ensures that whenever they are 1 you put it as 1. You are you are taking it as 1. So this constraint helps you in that you cannot put alpha i j equal to 0 whenever alpha i and

alpha j are both 1, together these 5 constraints will actually give you the same sort of objective function that you were doing earlier, but now in the terms of integer linear programming. So now, your integers alpha i's all the alpha i's and alpha i j's you have to choose such that this objective function is maximized and the constraints are satisfied. And then you can run your ILP solver to actually obtain the solutions.

So what is good about this approach is that it is very generic enough. So depending on your task you can define what is your objective function and what are your constraints. So for example, here we took very simple constraints independent of your task, but suppose in your task.

(Refer Slide Time: 22:40)



So for example, you are doing a summary from a say research article. And research article has various sections. Section 1, section 2, and section up to section 5, and now suppose you want to put a constraint that in my summary I want at least one sentence from each section. So you can easily put that constraint in your ILP.

So you can define what is the section information for each of the unit and then say for each section from it you will have at least one unit in your final solution. So that is a constraint that you can add. Similarly suppose you have some sort of data where that is temporal. So suppose news articles and you have over various days, so day 1, day 2, day n. So this generally happens when you are doing some sort of time line summarization. So you have about an event you have various articles in a timeline and you are trying to

summarize that. So there you want to put a constraint that I want, if a similar kind of information is there in 2 different time points. I want the later time point information or you can say that I want more information from the later time point than the previous time point.

Again you can put them as your constraints you define like you defined your length you can define the time period and you can say certain time period I want more than the previous time period. This can also add as your constraint. Then you can also change your objective function. So here we said we will sum over all the relevancy score of the textual units, but suppose you define, I want the I want to put some weight to the importance to the importance to the different series like sections or days.

So we can multiply the relevance score by certain weight. So this can be the importance weight of different days. So this can also come in your objective function. So like that this is very generic approach where you can keep on changing your objective function. So you can also maximize multiple things together and you can keep on adding your constraints depending on your task and you will still be able to find the solution using ILP.

So once we have done this. So we have talked about different methods for doing summarization. A graph based approach and an optimization based approach. So what were we discussing in summarization? Once we have selected the important sentences the next step will be how do I order these sentences. Now ordering might also be some sort of a heuristic approach. So what is the simplest approach that you will take. So once you have found the sentences from the document, you will order them in the same way in which they occur in the in the document. So although their relevance might be that the sentence at the bottom has the highest relevance, but you will take the sentence that is coming first in the in the document as the first sentence and so on. So the order in which they occur in the document you will provide that set.

But there are other approaches also. And they are called like you want to optimize the coherence of the summary. So that it becomes more readable. So what is one possible way? So you do not give anywhere drastically different sentences together. So you will say I will try to put them together in an order such that the 2 sentences that are close

enough are similar to each other. So that is one possibility. Or you can say that they talk about the same sort of main entities or events. So that is another criteria for ordering.

(Refer Slide Time: 26:42)

The next steps: Sentence Ordering

**Chronological ordering: the simplest method**  
List the sentences in the order, they appear in the document

**Coherence**

- Choose orderings that make neighboring sentences similar (by cosine)
- Choose orderings in which neighboring sentences discuss the same entity

**Topical ordering**  
Learn the ordering of topics in the source documents

Pawan Goyal (IIT Kharagpur)   Optimization Based Approaches for Summarization   Week 11, Lecture 2   7/8

So either you can list the sentences in the order they appear in the document. It is also called the chronological ordering. And you can optimize the coherence that is choose ordering that make the neighboring sentences similar by cosine similarity. Or choose ordering in which the neighboring sentences discuss the same sort of entity. And you can also do topical ordering. So you find out in document what topics are occur in what order and accordingly you put the sentences belonging to first topic will come first and so on. That is another possibility. So once we do that another approach another step could that I want to further reduce some sort of redundancy from there and that can be in the in the way we write.

(Refer Slide Time: 27:35)

*The next steps: Simplifying Sentences*

Parse sentences, use rules to decide which modifiers to prune

- Initial adverbials: For example, on the other hand, as a matter of fact, at this point, ...
- PPs without named entities: The commercial fishing restrictions in Washington will not be lifted unless the salmon population increases [PP to a sustainable number]
- Attribution clauses: Rebels agreed to talks with government officials, international observers said Tuesday
- Aappositives: Rajan, 28, an artist who was living at the time in Philadelphia, found the inspiration in the back of city magazines

Pawan Goyal (IIT Kharagpur)      Optimization Based Approaches for Summarization      Week 11, L1



So let us see what are the different ways in which you can simplify. So for simplification of sentences. So simplification is important because by simplifying sentences you can get more space into your final summary. So some sentences that contain some information that is not important you can remove that. And that is like not a very easy step. So what you will do? You will parse the sentences and remove certain specific clauses and phrases. Like for example, initial adverbials. So many sentences start with for example, on the other hand, as a matter of fact, at this point etcetera. All these are not important to convey the information and you can remove these initial adverbials.

Then some propositional phrases that do not contain the named entities can also be removed. So like here the commercial fishing restriction in Washington will not be lifted unless the salmon population increases to a sustainable number. To a sustainable number is a prepositional phrase and this may not be very important to convey the information here. Because it does not contain any named entity. So you can also remove this. Then you can also remove the attribution clauses like rebels agree to talk with government officials' international observers said Tuesday.

So this attribution to something this may not be important. So you can remove this also from your summary. Then certain appositives can also be removed like Rajan 28, an artist who was living at the time in Philadelphia found the inspiration in the back of city magazines. So the appositive here is an artist who was living at the time in Philadelphia

and this may not be important to convey the information. So like that you can also define many such rules. These rules can be again given manually to the system. And you can also learn these rules. If you have some sort of data that says what is the original sentence and how did some human simplify that. So from there you try to learn the rules that what kind of nodes in my parse tree can be removed and how I can simplify my sentences that is also possible.

So we discussed 2 3 different approaches of summarization. And how can you do some final polishing and post processing once you get the sentences to fit in a much smaller space. And as such there are many different approaches for summarization. You can always use these as your baselines, but you can also explore the other methods.

So in the next lecture we will briefly discuss that once you have done the summarization how do you go about evaluating your system summary. So you are getting some summary and how good is that. So what is the different methods we will talk about one specific method for that.

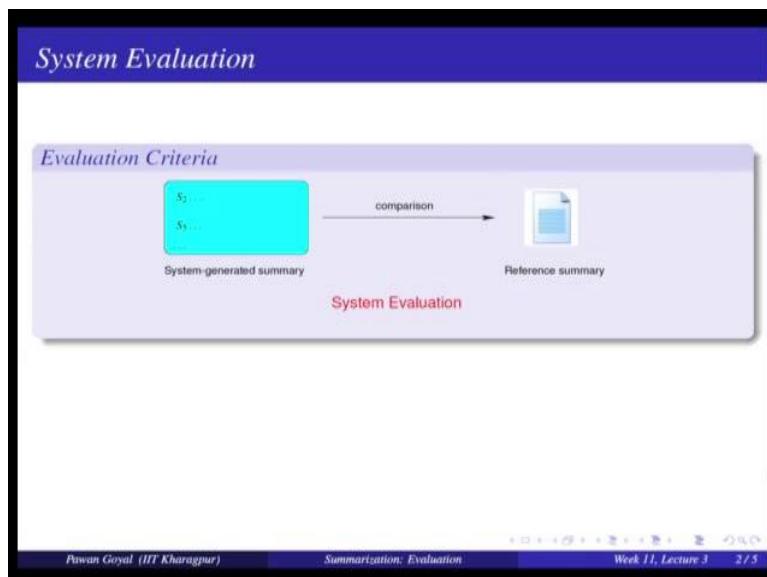
Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 54**  
**Summarization – Evaluation**

Welcome back to the third lecture of this week, so we have talked about different approaches of summarization; in this lecture we are talking about evaluation. So, how do we evaluate the summary that you have obtained using your system, so what do I mean by evaluation.

(Refer Slide Time: 00:33)



So, you run your summarization algorithm on your document set and then you are getting some set of sentences. So, as such assuming that you are running as extractive summarization; you will get a set of sentences as an output. You can also run different sort of abstractive summarization or whatever and you will get different sort of sentences as your output.

Now, how do I find out how good my summary is? So as such if you talk about, we will see what should be in criteria of evaluating a summary. So, we say if we give it to human, so human should say this summary contains the good amount of information from the original article. So, that is one criteria can be what is the informational coverage of this summary and then we can talk about how diverse the sentences are, so you are getting different sort of summary in here, you may talk about readability; if you are doing an abstractive

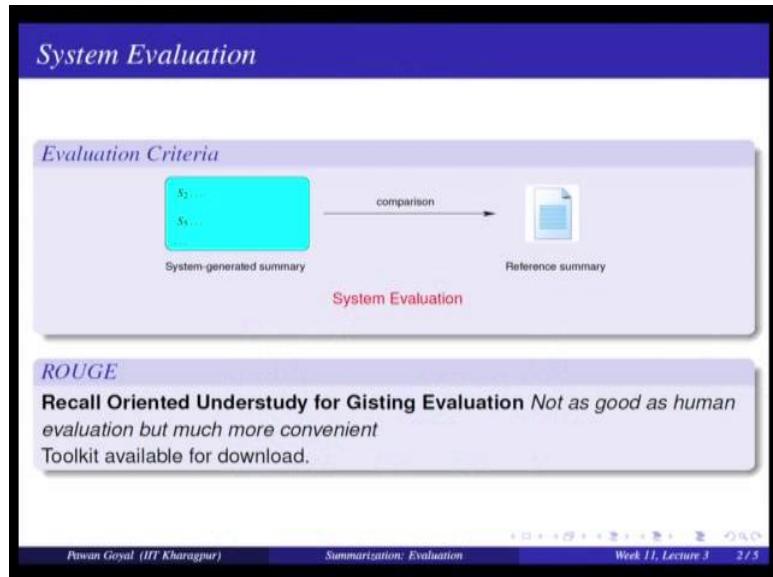
summarization and there can be some other criteria depending on your what is your task. So, are you getting the information that you wanted from this summary.

So, this you can do by some sort of human evaluation, you give the summary to some humans and ask them to rate the summary along these points from a scale say 1 to 5 and it is how good the summary is on the scale from 1 to 5 as per the readability, as per the information coverage as per the diversity and so on. But what we will see is there some method by which you can automatically evaluate; how good the summary is with respect to the document and there are some methods for doing that and one very popular method is using the Rouge score and that is what we will see; what is the idea?

So, your system is rating the summary for this document now in Rouge's method; you will assume that there are some humans that have created a gold standard summary for this document already. So, this helps in that you can now try out different different systems; get the summary and find out which one does the best with respect to the human. So, you have the human summary and you have some system generated summary and you want to compare these and this comparison is done by an approach; it is called the Rouge approach.

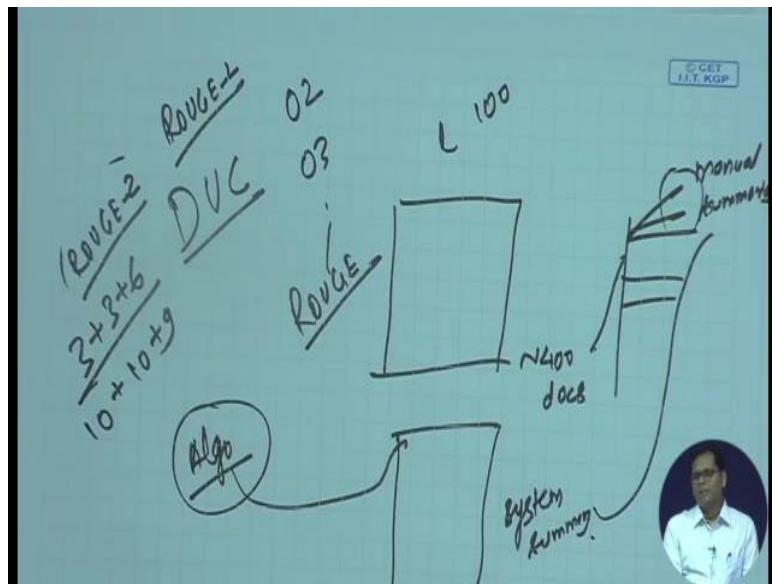
So, what do we have? So we have the system generated summary here, some sentences and a reference summary that is created by the human. Now in general; it may happen that you want to create multiple summaries for the same document, so the same document might have multiple reference summaries; so 3 reference summaries or 5 reference summaries still you should be able to compare this with the system generated summary.

(Refer Slide Time: 03:26)



So, the approach that is used is called the Rouge evaluation; recall oriented understudy for Gisting evaluation. So, as good as doing a human evaluation, but still it is quite convenient because for summarization, you have a lot of different computations and benchmark data sets. So, what you want to see that different systems as you propose different algorithms; are they able to improve on the previous approaches or not and a human evaluation may not be first of all feasible for doing for different algorithms, different variations and if it is there it may not be very reliable that do you get the same humans again and again and even if they are from different volunteers; how much they agree and all, so there can be many other issues with that. So, using an automated evaluation avoids all such problems and you say that for a benchmark data set. So, for summarization there are datasets like that are given in document understanding conferences.

(Refer Slide Time: 04:32)



So, they are by the name of DUC, so DUC has multiple datasets, so DUC 02, DUC 03 and so on. So there what do you have? You have some documents, so like they may be roughly 400 documents; for each document they provide some manual summary. So, this is again by using certain guidelines; they have built the manual summaries. So, they have done it once for all these 400 documents.

So, now once you have your algorithm; you again produce the summary for these 400 documents. So, this is your human summary, this is your system summary and then you try to see how close is the system summary to the manual summaries and this one; this thing you will do for. So, now, you can vary your algorithm and see I will find the how good this algorithm works on this dataset, how good the other variation works on this dataset and you can also compare on this benchmark, so that is why this automated evaluation is helpful.

Now, Rouge is one very popular method for doing this, so let us see how what is the basic idea for Rouge. So, Rouge again has many different variations, so that is do you want to find out similarity on only the unigrams; that is single words how much is single words are matching or bigrams; how many bigrams are matching or you want to go for longest common subsequence that you are matching and like that there are many variations. So, most popular ones are Rouge 1 and Rouge 2; how many unigrams are matching and how many bigrams are matching between the 2.

(Refer Slide Time: 06:26)

The slide has a blue header bar with the text "ROUGE for evaluation". The main content area contains the following text:  
Given a document  $D$ , and an automatic summary  $X$ :  
• Have  $N$  humans produce a set of reference summaries of  $D$  ( $N \geq 1$ )  
• Run system, giving automatic summary  $X$   
• What percentage of the n-grams from the reference summaries appear in  $X$ ?  
The formula for ROUGE-2 is displayed in a box:
$$ROUGE - 2 = \frac{\sum_{S \in \{RefSums\}} \sum_{bi-gram \in S} Count_{match}(bi-gram)}{\sum_{S \in \{RefSums\}} \sum_{bi-gram \in S} Count(bi-gram)}$$

At the bottom right, there is a circular video player showing a person speaking. The footer of the slide includes the text "Ptwan Goyal (IIT Kharagpur)", "Summarization: Evaluation", and "Week 11, Lecture 1".

So, what is the actual approach for doing that; so, suppose I have a document D and an automatic summary X; X is what my algorithm is providing. Now what I will do? I will have N humans produce a set of reference summaries. So, like I did in this (Refer Time: 06:43) dataset, so there are suppose 3 humans; they produce summary for each of this document. Now I have the automatic summary X, so what I will do? I will find out what percentage of the N gram from the reference summaries appear in X. So, this is like I am finding the record, how many N grams from reference summaries are appearing in the system summary.

So this N I can vary from 1 to n so on, so if I take n is equal to 1; this is the Rouge 1 measure; if I take n is equal to 2; this is the Rouge 2 measure. So, this is the example for using a Rouge 2 measure; that is you are taking bigrams. So, what you are doing? So you are counting; so your numerator is for all reference summaries; for all the bigrams that appear in the reference summaries; find out how many are matching with the system generated X; how many bigrams are there in the system generated summary x as well. So, you are counting that this is numerator; what is denominator, denominator is for all the reference summaries for all the bigrams count how many bigrams are there; that is you are finding out among all the bigrams that you can find in the all the summaries; how many are present in the human sorry system summary and immediately you can see that; if a bigram occurs in many of the human generated summaries, it gets a higher weight. So, if that bigram occurs in the system generated summary, it will start giving it a high weightage.

So, this is a very simple approach but this has shown to correlate a lot with the human judgments and this is one of the very popular method for evaluation, so this is well accepted method. So, let us try to see how this works on a simple example.

(Refer Slide Time: 08:33)

The slide is titled "ROUGE Example". It contains three main sections: "Reference Summaries", "System Summary", and "ROUGE-2".

- Reference Summaries:**
  - Human 1: water spinach is a green leafy vegetable grown in the tropics.
  - Human 2: water spinach is a semi-aquatic tropical plant grown as a vegetable.
  - Human 3: water spinach is a commonly eaten leaf vegetable of Asia
- System Summary:** water spinach is a leaf vegetable commonly eaten in tropical areas of Asia.
- ROUGE-2:**  $\frac{3+3+6}{10+10+9} = 12/29 = 0.413$

At the bottom, there are navigation icons and text: "Piwan Goyal (IIT Kharagpur)", "Summarization: Evaluation", "Week 11, Lecture 3", and "4 / 5".

So, suppose I have a document and there are 3 reference summaries provided by 3 humans. So, first summaries water spinach is a green leafy vegetable grown in the tropics, second one water spinach is a semi-aquatic tropical plant grown as a vegetable and third water spinach is a commonly eaten leaf vegetable of Asia; 3 different reference summaries and now your system produces this summary, water spinach is a leafy vegetable commonly eaten in tropical areas of Asia. So, what you will do? You will find out how many bigrams are common in human summary one and system summary plus common bigrams in 2 and summary plus common bigrams in 3 and summary that will be your numerator, denominator will be bigrams in 1 plus bigrams in 2 plus, bigrams in 3.

So, what are the bigrams in 3, so bigrams will be 1 less than the number of words, so because I can take bigrams like water spinach; spinach is a and so on. So, number of words are - 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, so there are 9 bigrams in third summary. Similarly here 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11; 10 bigrams, so my denominator here is 29; now what will be my numerator? So, numerator would be how many bigrams are common from here to each of these, so let us see system summary and human one water spinach is common 1 spinach is 2 is a 3 then system and human 2 water spinach 1; spinach is 2 is a 3 and then nothing else no other

bigram occur in both. So, 3 plus 3 and system and human 3 water spinach 1 spinach is 2 is a 3, then commonly eaten 4 then leaf vegetable 5 of Asia 6.

So, number of common bigrams are so 3 plus 3 plus 6 divided by total bigrams 10 plus 10 plus 9 that is the Rouge 2 score. Now this you will compute this you have computed for what this you computed for this one document, we will compute it for all the documents in your corpus. So, there is a toolkit available for Rouge, you can download that toolkit set up your system and then you will give all the documents; it will compute all the scores Rouge 1, Rouge 2, there is Rouge 1 and so on. There are many many other scores that you can compute and it can also tell you the confidence interval and all and then you can compare different approaches; how much they are the Rouge scores are different.

Now, one thing you must be careful while using the Rouge approach; in general when we are doing it for benchmark datasets, we have some constraint that I want a summary of length 100. But suppose when you are running your system some sentences are having, so you are stopping when the length is just above 100. So, suppose you are stopping at 100 and 500 and 700 and 8. So, if you want to use those summaries also there is a way in which in the toolkit you can provide this say that I have the length I should consider only length up to say 100 of a given system summary.

So, there is a parameter you can set in the toolkit that say I will take only up to 100 words will be taken everything else will be left out, so that is possible. So, by the same Rouge summary, you can choose different different lengths, you can also say that I want to do stemming. So, stemming will help in that words like boy and boys will be starting matching; if boy occurs in reference summary and boys occurs in the system summary; it will not match here but if you say I will do stemming then it will start matching. You can also say that I will remove the stop words before completing all this. So, that way you are only matching the non-stop words, not like is and a occurring in both that will not make sense.

So, there are many other options that you can see in the Rouge toolkit and they help you in getting a very good evaluation of your system. So that is what we saw here Rouge 2 score, if you compute for this these 3 summaries this particular system summary it will come out to be 0.413 and you can compute Rouge 1 and so on like this by this method.

So, there are many other methods for evaluation, so one other method is like pyramid evaluation. So, where the idea is that you define some sort of semantic content units from

your documents but they have to be manually defined and that is why it does not become completely automatic, if you have a large corpus you have to define by on your own some semantic content unit and then your idea will be whatever summary you are getting, you should have the maximum coverage of those semantic units in a nice, so this is a nice method also but it is not it requires you to define these units and there are other approaches that have seen that. So, given the document and the summary can you find some property among these that can tell you whether this is a good summary or not but still I would say you can always rely on Rouge 2. If you have a ground truth summary if you do not have ground truth summary one way is you create your own summary or you do you go for some human evaluation if you do not have a lot of documents to evaluate.

So, that finishes our discussion on summarization, but so there are certain things that we did not discussed. So, this is something that we talked in the initial one of the initial slides that what are the different genres of summarization. So, we have focused our whole discussion on single document summarization and generic summarization and extractive summarization, what we were doing taking the sentences extracting the sentences but what about doing a multi document summarization.

So, multiple document summarization is not very different from what we are seeing here, except that now you are having different documents. So, as such you can apply the same algorithm like lexing algorithm where, you take all the sentences in all the documents and apply your algorithm but there once you have the summary you might want to give different weightage to different documents. Suppose you know one document is more important than another document, you will have more weightage for that sentences from that document. Also the similarity between the sentences in same document might be given different sort of function and similarity across documents, you might also use how many how similar two documents are and so on. So, there are many different tricks that you can apply there but overall the idea is roughly the same, so it would not be very very different.

(Refer Slide Time: 15:43)

*Further Discussions*

- Multi-document summarization
- Query-specific summarization

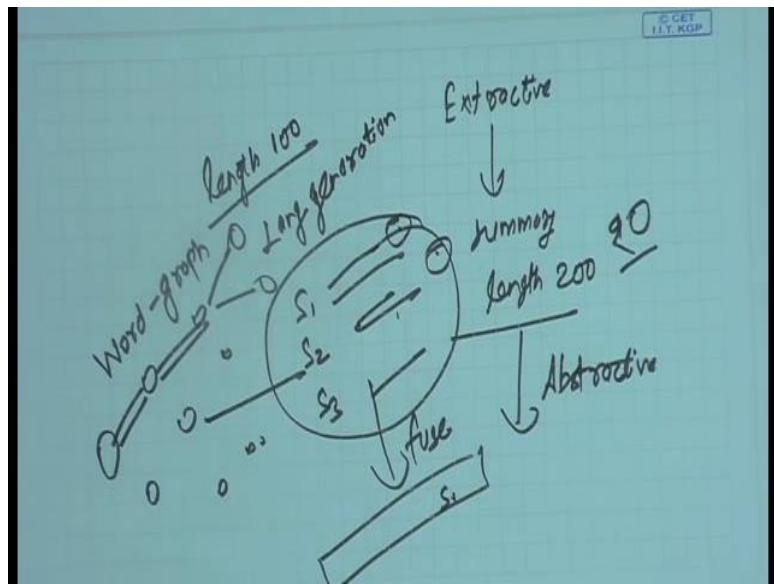
Ptwan Goyal (IIT Kharagpur)      Summarization: Evaluation      Week 11, Lecture 3      5 / 5

Then there is something like a query specific summarization; that is user has given a query and I want a summary with respect to that query. So, one particular approach would be first you find out what are the sentences that are similar to that query and then you summarize only on those that way reduce your initial set or it can be first you get a larger summary than what you wanted and then you apply your; you find out which ones are more closer to the query then you retain only that.

So, first find out the important sentences from there retain only those that are closer to the query and there can be some other methods also. So, idea is and here again you can do a query specific multi document summarization. So, you have a query, you have a whole repository; first find out using some information to the method which of the documents are relevant to the query, so this becomes your multi document set from there using your summarization and pick those sentences that are relevant to your query. So, this is like you can apply query especially summarization by using same sort of techniques but some preprocessing some post-processing based on the query.

And then finally, you have the abstractive summarization, now abstractive is slightly trickier. So, because here now you are not just worried about just picking up the sentences, you are taking the sentences if some sentence contains similar information you are trying to fuse them together. So, what is in general done is that whenever you want to produce abstractive summary, you apply the extractive summary first get a larger set.

(Refer Slide Time: 17:19)



So, that is suppose I want a summary of length 100 I want a summary of length 100, so what I will do? I will apply extractive summarization get a summary of say length 200. So, twice more than what I need; now I apply some abstractive method in abstractive method in general what I will do. So, here I will take; suppose I have here 10 sentences 10 or say 20 sentences, I will find out which sentences are very similar or close to each other. Suppose I find S1, S 2 and S 3 are similar to each other.

So, what I will do? I will take these sentences and try to fuse them together to get a single sentence; then how this fusion will take place? I will say are there some words that are common here and there and then some information that is providing some information that this is providing. So, I will take the common part take information from here and here. So, there are many approaches for doing that many of them are based on a word graph.

So, that is you construct all the words, take all the words in the sentences, construct a graph and then you try to follow the path, these words are connected, these words are connected in the second sentence also these words will be connected, but there will be some diversion wherever there is diversion you try to take both of these together, by doing some sort of generation language generation. So, that is like you can connect them by and, or some other different connectors but whatever is common you will take it only once.

So, there are again, this is again a nice area where lot of work is happening that how do you construct the abstractive summarization. But again the idea is that you first take the extractive

one and then create an abstractive from there and again here you can use your IIP based method to find out what paths to be taken, paths can be based on the unigrams bigrams or higher length. So, with that I will say that this finishes our discussion on summarization and this is a very nice application lot of work has been done and is being done right now in this field, there are lot of different applications in.

So, you can talk about news summarization and scientific article summarization at the same time you can talk about summarization from your feeder streams, quora answers, stack flow, stack overflow answers and so on. So, lots of tweets are there for certain events you want to summarize those from a disaster event you want to summarize those. So, that way you can use your extractive summarization abstractive summarization to also be able to help you.

So, in the next lecture we will start a new application topic that is text classification again that is very very important and we will discuss one very again very appropriate baseline for that.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 55**  
**Text Classification – I**

Welcome back to the fourth lecture of this week, so we have been talking about various applications and we finished our discussions on text summarization. So, in the last two lectures of this week we will focus on one very important application in NLP that is called text classification. So, we will discuss what are the different applications were this whole approach to text classification can be used and we will talk about one simple baseline that is how do you use a Naive Bayes classifier for text classification and we will also spend some time in discussing; how do you evaluate your models. So, once you have built your system and how do you evaluate your system.

So, starting with what do we mean by text classification, what are some of the examples you can think of. So, text classification as the name would say, so you are given a piece of text; it can be a sentence, it can be a document, it can be a paragraph or any other unit of text and even to classify it into certain categories and the categories can depend on what is your application and that is where this becomes a very very generic problem because many of the problems in the NLP, you can treat them as text classification problems. We will see some examples.

(Refer Slide Time: 01:30)

*Example: Positive or negative movie review?*

• unbelievably disappointing  
• Full of zany characters and richly applied satire, and some great plot twists  
• this is the greatest screwball comedy ever filmed  
• It was pathetic. The worst part about it was the boxing scenes.

Pawan Goyal (IIT Kharagpur) Text Classification - I Week 11, Lecture 1

So suppose the problem is you are giving a movie review and you have to find out is it a positive review or a negative review. So, here are some examples, so this is the field of sentiment analysis. So, you have different reviews here like unbelievably disappointing and this you would say immediately; this is a negative feeling, then full of zany characters and richly applied satire and some great plot twists and we will say this is a positive review, this is a greatest screwball comedy ever filmed again positive review it was pathetic and so on this becomes a negative review.

So, now the problem here is, suppose you are given a lot of movie reviews or say product reviews or hotel reviews and we have to find out each and individual sentence is it talking about some positive sentiments for this product or negative sentiments for the product, how do I do that automatically. So, this becomes a classification problem, so given a text I want to classify it into one of these classes; positive, negative or neutral.

(Refer Slide Time: 02:32)

The slide has a blue header bar with the text "Example: Male or Female Author?". The main content area contains two numbered lists:

1. By 1925 present-day Vietnam was divided into three parts under French colonial rule. The southern region embracing Saigon and the Mekong delta was the colony of Cochinchina; the central area with its imperial capital at Hue was the protectorate of Annam...
2. Clara never failed to be astonished by the extraordinary felicity of her own name. She found it hard to trust herself to the mercy of fate, which had managed over the years to convert her greatest shame into one of her greatest assets...

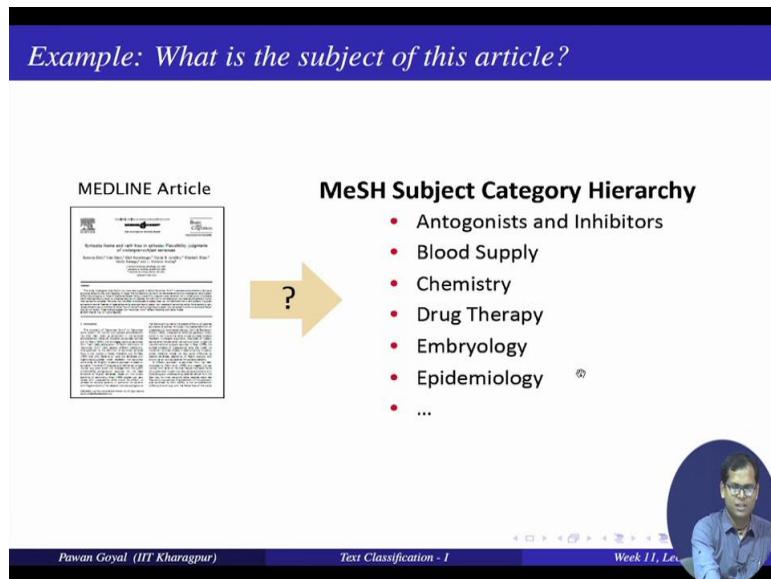
At the bottom of the slide, there is a navigation bar with icons for back, forward, search, and other presentation controls. The footer of the slide includes the names "Pawan Goyal (IIT Kharagpur)" and "Text Classification - I", the week information "Week 11, Lecture 4", and the slide number "3 / 14".

Let us take another example, so here this is about authorship attribution. So given a piece of text; find out certain demographics of the author. So, what is the gender of the author is it a female author or the male author, what is the age of the author, is he in the early 20's or 30's or is he a very old author and there can be many other things like what is the country that author belongs to and then you can also go to very very fine label to saying who is the author for this particular paragraph or text. Now why do you want to treat as a classification problem? So you will say that authors for a particular demography will have certain traits. So, they will have a particular style of writing, so can I try to seek given a text what style of writing it matches and match to the corresponding class.

So, here for example there are two pieces of text and the problem is whether this is a male author or a female author and then you can use certain heuristics and some sort of studies that people have done that when they are male authors, they will write lot of about lot of different facts and then female authors; they will be some certain opinions; opinions will be in majority. So, when you see these two piece of text and you will say the first one is talking about lot of different facts about the year, information and the place information and the second text is about certain opinions; certain subjective information and then this might be used to say which one is the male author, which one is the female author.

But in general when you are given a lot of data about writings from male and female author, you can try to learn a model that can classify a piece of text; is it from a male author or a female author.

(Refer Slide Time: 04:21)



Similarly so here you have a research article like Medline article and it can be from any research domain and what is the problem? For that, suppose for your digital library; you have a set of categories or broad topics and given a new article, you want to assign it to one of these topics. So, that is putting this in somewhere in your hierarchy of categories, so here for example, the categories can be antagonists and inhibitors; blood supply, chemistry, drug therapy, embryology, epidemiology. So, all these are different categories that you have in your mesh hierarchy; given a scientific article you want to put it in one of these categories. So, this is again a classification problem all these categories are your classes and given a piece of text, you want to find out what is the category it matches to.

So, immediately start seeing that this text classification is a very very wide problem and different, different applications you can always convert them to some sort of text classification problem.

(Refer Slide Time: 05:25)

- Assigning subject categories, topics, or genres
- Spam detection
- Authorship identification
- Age/gender identification
- Language identification
- Sentiment analysis
- ...

Pawan Goyal (IIT Kharagpur) Text Classification - I Week 11, Lecture 4 5 / 14

So, like assigning subject categories, topics or genres this is one category of text classification; then it can be spam detection, you are given an email or it can be now tweets, movie reviews or product reviews, is it spam or not. So, it is classification problem is it spam or not spam then authorship identification, so we talked about this; who is the author, find out the demographics of the author and so on, age and gender identification, language identification so that is you are given the piece of text, find out what is the language it belongs to.

So, here what is important is that when you are, so for example, let us talk about English versus Hindi. So, whether language is English or Hindi, so what do you think is it a simple problem or difficult problem? So, suppose you are writing Hindi in Devanagari then this basis on the script, you can find out whether its English or Hindi, but when you type Hindi in your say in your comments, Facebook posts or twitter. So, do you write in Devanagari or do you mainly use the roman script, you mainly use the roman script and that is we use the transliteration for writing in Hindi.

So that means, when have the English text and the Hindi text they have the same script. So, script is same; that means, you have to go to the actual word level to find out is it coming from Hindi or English and that is where the problem starts, the same words can be written both in English and Hindi, so same way For example, take pure in English and pure in Hindi; they will have roughly the same transliteration P u r e. So, given this word, it will be difficult to find out if it is a English or Hindi word, so like that there will be different problems and given a piece

of text; what is the language there you need to build up some different language specific models. Then there is a field of sentiment analysis, that is what we talked about all the people give an entire week for talking about opinions and sentiment analysis that will be the last week of this course.

(Refer Slide Time: 07:46)

The slide has a dark blue header bar with the title "Text classification: problem definition". Below the header, the slide content is organized into two main sections: "Input" and "Output".

**Input:**

- A document  $d$
- A fixed set of classes  $C = \{c_1, c_2, \dots, c_n\}$

**Output:**

A predicted class  $c \in C$

At the bottom of the slide, there is a navigation bar with icons for back, forward, search, and other presentation controls. The footer contains the text "Pawan Goyal (IIT Kharagpur)", "Text Classification - I", "Week 11, Lecture 4", and "6 / 14".

So, let us formally define this problem of text classification, so what do you have? You are given a document, by document I mean a piece of text that is the unit that you want to classify. So, in general it can be a sentence paragraph or whatever, so you have a document  $d$  and you have fixed set of classes  $C$ ;  $C$  equal to  $C$  n  $C = \{c_1, c_2, \dots, c_n\}$ . So, this document  $d$  you want to classify it into one of these  $n$  classes of course, there can be some variations where you may want to assign it to many of the classes or so we will talk about it, but let us take the simplistic assumption that each document belongs to one and only one class. So, given a document I want to assign it to one of these classes, this is a text classification problem.

Now, so my output will be one of the classes, one of the class from all these, from this set of classes.

(Refer Slide Time: 08:45)

**Classification Methods: Hand-coded rules**

- Rules based on combinations of words or other features

**Spam**  
black-list-address OR ("dollars" AND "have been selected")

**Pros and Cons**  
Accuracy can be high if rules carefully refined by expert, *but building and maintaining these rules is expensive.*

Pawan Goyal (IIT Kharagpur)    Text Classification - I    Week 11, Lecture 4    7 / 14

So, what can be the simplistic method that I can use, so like many other application that we have seen. So, we can also use some hand-coded rules, so let us say you have the problem of spam detection, you have an incoming email and you want to classify into spam or not a spam. So, what will be the simplistic model, so simplistic model could be, try to see some spam emails and make some hand-coded rules. So, what is the common spam that you remember? So it is like you have been selected for some: 100 million dollars or 750000 dollars and so on, this is a very common form of a spamming email. So you can have some hand-coded rules like; if the email is coming from some blacklist addresses. So, suppose you have a list of blacklist either URLs and email ids. So, if it is coming from there or if it contains something like dollars and have been selected, so then it is a spam.

So, this is one form of hand-coded rule that you can build by seeing what are the different spams; that email that I see regularly and like that you can try to find out many other spams and build these various rules; that is one approach.

So, like the other application that we have seen, so what are the pros and cons for using hand-coded rules? So, pros are that if you are an expert and you can write very good rules, then in general they will be very very precise, they will be quite accurate, so you will get a high precision. So, whenever you have something like dollars and have been selected you will probably be; it will probably be a spam email, but other problem is that you do not know how many such rules you have to build. So, if you do not have sufficient rules, you will not have a

very high recall, so you will not be able to find out all the spams. So, we will talk about these evaluation measures precision spams also in detail during this topic. So, maintaining is also quite expensive.

(Refer Slide Time: 11:01)

Classification Methods: Supervised Machine Learning

- Naïve Bayes
- Logistic regression
- Support-vector machines
- ...

Pawan Goyal (IIT Kharagpur)      Text Classification - I      Week 11, Lecture 4      8 / 14

So if you do not want to use hand-coded rules; what is the other option? So, you will use some supervised classification method and so some very popular methods are like Naive Bayes classification, logistic regression that is similar to what we talked in terms of maximum entropy model. So, what is the probability of a class given an input and a history  $x$ , so you will say this is linear function over the various features;  $\sigma(\lambda_i f_i)$  and then you have an exponent term; exponent of summation  $\lambda_i f_i$ .

So, this is a Maxent model or logistic regression model, but you can also use other models like Naive Bayes models and support vector machines, SVM and all these require thinking of various features that would be helpful for this particular task. So in these two lectures, we will focus on one model that is Naive Bayes model and that is one of the very powerful baselines for text classification. So, this is the default model that you would use for any text classification task, so we will see what is the Naive Bayes model and we will also do a working example for this.

(Refer Slide Time: 12:15)

- Simple classification method based on Bayes' rule
- Relies on very simple representation of document - Bag of words

Pawan Goyal (IIT Kharagpur)    Text Classification - I    Week 11, Lecture 4    9 / 14

So, what is the Naive Bayes model? So that is; it uses Bayes rule to do classification of text into certain classes and this relies on a very simple representation of documents that is bag of words and by now you understand what is bag of words model that is; you are given a piece of text, now in bag of words model the order in which the words occur in the document do not matter, what matters is that what all words are there. So, it is like a set of words; that is the bag of words assumption, now what are the other assumption that Naive Bayes model uses; let us see that model in detail.

(Refer Slide Time: 12:57)

Test document

?

Machine Learning	NLP	Garbage Collection	Planning	GUI
learning	parser	garbage	planning	...
training	tag	collection	temporal	
algorithm	training	memory	reasoning	
shrinkage	translation	optimization	plan	
network...	language...	region...	language...	

Pawan Goyal (IIT Kharagpur)    Text Classification - I    Week 11, Lecture 4    10 / 14

So, let us quickly see what do I mean by bag of words model for document classification. So assume that you have certain documents with you and you know also the category, so suppose you are talking about certain research domains and you want to put all your documents or scientific articles into one of these domains. For example, here you have an article from machine learning, another from NLP then garbage collection, planning GUI and so on.

Now these documents will contain certain words, so what is bag of words model? So, you assume that document is nothing, but this bag of words. So, here all these words that occur in the document; corresponds to this document learning, training, algorithm, shrinkage, network etcetera; they all this is all the document and then this is assigned a category of machine language. Similarly here parser, tag, training, translation, language these are all the words that occur in the document and document is assigned the category of NLP and so on for all these documents.

So, now what is your task? So you know there will be some data that is already labeled with these classes; at test time you will get a document but you do not know what is the class for this document. So, for example, at test time you get a document with these words parser, language, label, translation and you do not know what is the class of this document; among all these possible categories. So here based on what are the words that are occurring in this document, you want to assign it to one of these classes and there you are using the bag of words assumption that I know what are all the words that are there and I do not care about the order information.

(Refer Slide Time: 14:48)

Bayes' rule for documents and classes

For a document  $d$  and a class  $c$

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

Pawan Goyal (IIT Kharagpur) Text Classification - I Week 11, Lecture 1

Now once I have the scenario, how do I use the Naive Bayes model for classification? So, all of you know the Bayes rule by now, so what is the Bayes rule.  $P(c|d) = (P(d|c)P(c))/P(d)$

(Refer Slide Time: 15:00)

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$
$$C_{MMP} = \arg \max_{C \in C} P(c|d)$$
$$\prod_{x_i} P(x_i|c)$$

Naive assumption

$$\arg \max_C P(d|c)P(c)$$
$$\arg \max_C P(x_1, \dots, x_n|c)P(c)$$

features

So, it would say that probability of a class given a document would be; so how do you write it;  $P(c|d)$  given  $d$  probability of the class given a document is nothing, but probability  $d$  given  $c$   $P(d|c)$  given  $P(c)$ . So, now why are we using the Bayes rule here,  $c$  is a particular class and  $d$  is a document. Now I want to find out the probability for all the different classes given this test document and I want to take the one that is having the maximum probability. So, my problem

is take the class that is giving the maximum probability argmax;  $c$  in the set such that probability  $c$  given  $d$  is the highest and you can also give the maximum of posterior probability; this is the class that you get by Bayes rule.

Now so the question is how do you compute this probability; probability of a class given  $d$  for all the different classes. Now what is the Naive Bayes model, so if you remember we talked about two different families of model; one was generative models, another was discriminative models, so Naive Bayes is a kind of generative model. So, that is it will say that you have the class first and then you generate different documents for that class. So, what would happen in the model? So you have the class  $C$  and then you are generating the document  $d$  from this class.

So, that is suppose I want to generate a review; it says that first you think that whether you want to generate a positive or negative review. So, class will come first and then for that class you will generate the document, so that is how the probabilities will flow. So, you can find the probability with document given the class from the generative model. So, that is why you cannot compute this directly, so you have to compute probability  $d$  given  $C$ . So, that is where the Bayes rule comes into picture, so I want to convert that into probability  $d$  given  $c$  and how do I do that using the Bayes rule. So, this becomes argmax over all classes; probability  $d$  given  $c$ , probability  $c$  given probability  $d$ .

And now because  $P d$  is common for all the classes; yes you know the given document, so this you can remove. So, this is the probability that I have to estimate argmax over  $c$  probability  $d$  given  $c$ , probability of  $c$ ; what is probability of  $c$ ? That is the prior probability of the class, how much this class is prevalent in this collection, this is if I do not know about this document, what can I say about which class is more probable than other this is the prior probability and then this tells you given this data; when you have seen the document then what is the probability of document, getting generated from this class. Together they give you the posterior probability of class given the document. So,  $P c$  you can easily compute; if you are given a training data, you can find out how often this class  $c$  occurs among all the classes.

Now, how do you compute probability  $d$  given  $c$ ? Now for that you have to see what is your document, your document is nothing but bag of words. So, suppose the document contains some words  $x_1$  to  $x_n$ . So, then you can convert  $d$  to  $x_1$  to  $x_n$  and you can write like; this is same as argmax over  $c$  probability  $x_1$  to  $x_n$ ; given  $c$  probability  $c$  and where  $x_1$  to  $x_n$  are various. So, in general you can call these as various features, if you are only using your words

then features will be only words, if you are using some other information like it may be the time of the document or the length of the document and many other things then this is also possible here.

So, this is that you get by this simply using the Bayes rule. So, now you have this, so again this one is easy, but this might be difficult to compute; what is the probability of getting all these features together given this class and that is where you make an Naive Bayes assumption. So, this is the naive assumption that is why this is called the Naive Bayes model. So, what is the naive assumption here? So that is the probability of all these features given the class can be written as; so this whole thing can be written at multiplication of probability  $x_i$  given  $c$ ; for all  $x_i$ .

So, that is each feature is conditionally independent of each other given the class. So, this becomes; this simplifies this model to a very large extent. So, now you have to worry about the joint probability of the all the features, you can talk about the individual probability of different features and then you multiply. So, this is from the naive assumptions and that is where you are using the Bayes rule and together that is why this is called the Naive Bayes model.

So, you know what is the naive assumption here and you are using the Bayes rule for computing this probability.

(Refer Slide Time: 20:43)

*Bayes' rule for documents and classes*

For a document  $d$  and a class  $c$

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}$$

*Naïve Bayes Classifier*

$$\begin{aligned} c_{MAP} &= \arg \max_{c \in C} P(c|d) \\ &= \arg \max_{c \in C} P(d|c)P(c) \\ &= \arg \max_{c \in C} P(x_1, x_2, \dots, x_n|c)P(c) \end{aligned}$$

Pawan Goyal (IIT Kharagpur)      Text Classification - I      Week 11, Lec 1

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(c|d) \text{ where } c \in C$$

$$= \operatorname{argmax} P(d|c)P(c)$$

$$= \operatorname{argmax} P(x_1, x_2, \dots, x_n|c)P(c)$$

So, now if we go back, so we want to compute the find out the class that gives the highest probability; probability c given d and then we write using Bayes rule in this form and then we say document d is nothing, but all the features that are there in the document, so this is collection of the features x 1 to x n given the class c.

(Refer Slide Time: 21:02)

### Naïve Bayes classification assumptions

$P(x_1, x_2, \dots, x_n|c)$

*Bag of words assumption*  
Assume that the position of a word in the document doesn't matter

*Conditional Independence*  
Assume the feature probabilities  $P(x_i|c_j)$  are independent given the class  $c_j$ .

$$P(x_1, x_2, \dots, x_n|c) = P(x_1|c) \cdot P(x_2|c) \dots P(x_n|c)$$



Pawan Goyal (IIT Kharagpur)      Text Classification - I      Week 11, Lecture 1

Then how do you compute probability x 1 to x n given the class c? So, there you are making two assumptions one is bag of words that is the position of a word in a document does not matter. So, x 1 to x n whatever order they occur is immaterial, so I am only talking about a set of words here and then what is the other conditional independence, the naive assumption that the feature probabilities x i given c j are conditionally independent of each other; given the class c j. So, I can write probability x 1 to x n given c as probability x 1 given c; times probability x 2 given c up to probability x n given c.  $P(x_1, x_2, \dots, x_n|c) = P(x_1|c).P(x_2|c) \dots P(x_n|c)$

So, now I have taken all these assumptions and now I have a very simplified model that is I will take the class that gives the maximum argmax over P c times P x given c and x here are all my different words. Now the next question is how do I compute these probabilities from my

training data, so how would you compute probability c. So, P c would be; what is the probability of this class in the training data.

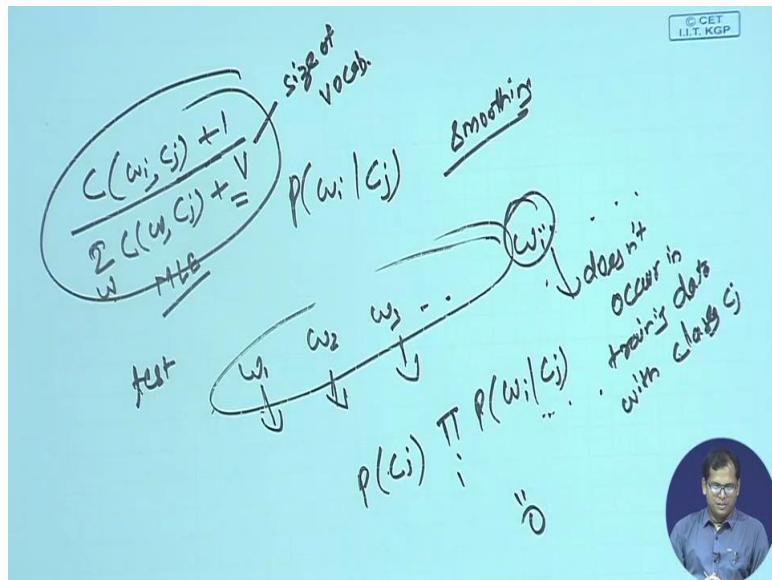
(Refer Slide Time: 22:20)

So, that is in my training data how many documents have been labeled with this class. So, this is doc count c is equal to c j i; how many documents in my training data labeled with this class divided by how many documents do I have in my training data. So, this is what is probability of this class in my training corpus. Then how do you compute probability x i given c j; probability of a word given with the class and that will again be very easy similar to what we did in the case of language modeling.

So, number of times this word occurs in the class divide by number of all the words that occur in that class; that is number of times this word is occurring w i occurring in class c j divided by count of all the words occurring in class c j, this can also be taken as the whole corpus count of class c j; how many unique not unique, how many different total tokens are there in this class c j. So, this will give me both the probabilities and so this I can estimate from my training data. So, I have these probabilities, I can multiply these and get the best class for a given test document.

So, what can be one problem with this approach only taking these probabilities as such? So, assume that in training data all the classes occur at least once, so the prior probability of a class will not be 0.

(Refer Slide Time: 23:58)



But what about this particular term probability  $w_i$  given  $c_j$ . Suppose you are given a test document and test document I mean there are different words  $w_1, w_2, w_3$  and so on. There is a word  $w_i$  that does not occur in training data with class  $c_j$ , but all the words they occur some number of times with class  $c_j$ . So what will happen with my estimate of the posterior estimate? So this will become probability of  $c_j$  times, probability  $w_i$ ; given  $c_j$  for all  $i$ .

Now while all the other  $w_i$  are having a positive probability, this one is giving me 0 probability. So, I multiply this everything becomes 0; irrespective of even if the other words were very highly suggesting that class  $c_j$ . So, only one word if it does not occur in that class, it will make this whole probability go to 0 so that means, we need to do something to avoid this scenario and what should be that we will be doing and so that will again remind you are what we did in the case of language model to avoid the 0's.

So, we do something like a smoothing, so we use the smoothing to get these counts, to get these probabilities. So, how do I use smoothing? So till now I am computing probability  $w_i$  given  $c_j$  as, right now this probability is computed as count of word  $i$  in class  $c_j$  divided by summation over count of any word  $w$  in class  $c_j$ ; for all words, that is the current probability that is the MLE; maximum likelihood estimate. Now how do you use smoothing? Suppose I use the simplest smoothing that is add one smoothing. So, I will now write it as plus 1 in the denominator I have to add 1 to all the  $w$  words, so this will be plus  $V$ ; where  $V$  is my size of

vocabulary and this will be now add-1 smoothing and that is what I will be using. So, once I do that this will have some small probability 1 by V and this whole thing will not go to 0.

So that is what is the problem with using maximum likelihood estimate, so suppose in your training data, we have not seen the word fantastic in class positive. So, you know this probability will be 0; probability of fantastic given positive will be 0; while other words in the review or in the sentence might indicate positive.

(Refer Slide Time: 27:04)

*Laplace (add-1) smoothing*

$$\hat{P}(w_i|c) = \frac{\text{count}(w_i, c) + 1}{\sum_{w \in V} (\text{count}(w, c) + 1)}$$

$$= \frac{\text{count}(w_i, c) + 1}{(\sum_{w \in V} (\text{count}(w, c)) + |V|}$$

Pawan Goyal (IIT Kharagpur)    Text Classification - I    Week 11, Lecture 4    14 / 14

$$\hat{P}(w_i|c) = \text{count}(w_i, c) + 1 / \sum_{w \in V} (\text{count}(w, c) + 1)$$

$$= \text{count}(w_i, c) + 1 / (\sum_{w \in V} (\text{count}(w, c)) + |V|$$

So, this whole thing will go to 0, so instead we use a; add-1 smoothing. So, count plus 1 divided by summation over count w c plus 1; that will give me a plus 1 in the numerator and plus V in the denominator and that is my add-1 smoothing.

So, we talked about the Naive Bayes model and how do we compute various probabilities and we saw one problem with the probabilities; that some probabilities might go to 0 and that will take the whole thing to go to 0. That is why we can do n simple add-1 smoothing, so now in the next lecture, we will take an example and see how do we actually compute all these different

probabilities from a simple training data and we find the probability for the test sentence or document.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 56**  
**Text Classification – II**

Welcome back for the final lecture of this week. So, we had started talking about text classification and we discussed the naive Bayes model for text classification and today we will take a working example for how do we use naive Bayes model and then we will discuss various other issues with classification and also talk about the evaluation of classification.

(Refer Slide Time: 00:41)

A worked example							
	Doc	Words	Class				
$\hat{P}(c) = \frac{N_c}{N}$	Training 1	Chinese Beijing Chinese	c				
	2	Chinese Chinese Shanghai	c				
	3	Chinese Macao	c				
	4	Tokyo Japan Chinese	j				
$\hat{P}(w c) = \frac{\text{count}(w,c)+1}{\text{count}(c)+ V }$	Test 5	Chinese Chinese Chinese Tokyo Japan	?				
<b>Priors:</b>							
$P(c) = \frac{3}{4}$		<b>Choosing a class:</b>					
$P(j) = \frac{1}{4}$		$P(c d5) \propto 3/4 * (3/7)^3 * 1/14 * 1/14 = 0.0003$					
<b>Conditional Probabilities:</b>							
$P(\text{Chinese} c) = (5+1) / (8+6) = 6/14 = 3/7$							
$P(\text{Tokyo} c) = (0+1) / (8+6) = 1/14$							
$P(\text{Japan} c) = (0+1) / (8+6) = 1/14$							
$P(\text{Chinese} j) = (1+1) / (3+6) = 2/9$							
$P(\text{Tokyo} j) = (1+1) / (3+6) = 2/9$							
$P(\text{Japan} j) = (1+1) / (3+6) = 2/9$							

Let us take an example. So, what you are seeing here? So your training set that contains 4 documents document 1 to 4 and so document contains some words. So, like 1 contains Chinese Beijing Chinese and so on and there are some labels assigned. So, this document corresponds to class c that might be Chinese and this corresponds to class j that can be Japanese.

And then in test data you have 1 document; document number 5 that contains Chinese, Chinese, Chinese, Tokyo, Japan and you want to assign it to one of the class; one of these 2 classes. So, how do we solve this problem using naive Bayes model? So, remember what is a naive Bayes model? We need these 2 probabilities  $\hat{P}(c)$  and  $\hat{P}(w|c)$  for all these 3 words. So, let us try to compute these probabilities.

(Refer Slide Time: 01:38)

The image shows handwritten notes on a light blue background. At the top right is a small logo with the text "© CET I.I.T. KGP". The notes are organized into two columns separated by a vertical line.

**Column C:**

$$\hat{P}(c) = \frac{3}{4}$$

$$P(\text{Chinese}/c) = \frac{c(\text{Chinese}, c) + 1}{n(c) + |V|}$$

$$= \frac{5+1}{8+6} = \frac{6}{14}$$

**Column j:**

$$\hat{P}(j) = \frac{1}{4}$$

$$P(\text{Chinese}/j) = \frac{c(\text{Chinese}, j) + 1}{n(j) + |V|}$$

$$= \frac{1+1}{3+6} = \frac{2}{9}$$

**Below the columns:**

$$\frac{\hat{P}(c) \cdot P(\text{Chinese}/c)^3 P(\text{Tokyo}/c)}{P(\text{Japan}/c)}$$

$$\frac{\hat{P}(j) P(\text{Chinese}/j)^3 P(\text{Tokyo}/j)}{P(\text{Japan}/j)}$$

Test Chinese - 3, Tokyo - 1, Japan - 1

$$P(\text{chinese}/c) = (c(\text{chinese}, c) + 1)/(n(c) + |V|)$$

$$P(\text{chinese}/j) = (c(\text{chinese}, j) + 1)/(n(j) + |V|)$$

I have class c i have class j. So, I need to find out what is  $\hat{P}(c)$  and  $\hat{P}(j)$ . So, what will be  $\hat{P}(c)$ ? That is the probability of class c in maintained data. So, it occurs 3 times out of 4. So, this will become 3 by 4 and this will become 1 by 4 then I need to compute different probabilities for words. So, what are the words here in my test documents? I have Chinese 3 times then Tokyo once, Japan once.

These are document in my test data; test document, I want to assign it to some class and what will be the probability of these 2 classes? The probability of class c would be  $\hat{P}(c)$  times probability Chinese given c to the power 3 because it is agreeing 3 times probability Tokyo given c probability Japan given c and this will be  $\hat{P}(j)$  probability Chinese given j to the power cube probability Tokyo given j and probability Japan given j. So, now, I already know this and this. So, I need to compute the other 3 probabilities. So, how do we compute probability Chinese given c? This would be number of times the word Chinese occurs with this class, count of Chinese with c and I am using (Refer Time: 03:41) smoothing. So, it will be plus 1 divide by all the words that occur in the class Chinese. So, I can call it number of words in class Chinese plus my vocabulary size.

Now, what is the vocabulary size here and what is  $n_c$ ? Let us see, similarly you can compute

probability Chinese given j as count of Chinese in class j plus 1 divide by number of words in class j plus my vocabulary size. So, let us see from the documents how many times the word Chinese occurs in class c. So, it occurs 1, 2, 3, times actually 1, 2, 3 and 4 times. In class j, it occurs once. What is the vocabulary size of Chinese 1 2 3 not vocabulary size, how many different tokens are there in class Chinese? 1, 2, 3, 4, 5, 6, 7, 8, in Japanese 1 2 3 and vocabulary size how many unique words? 1 Chinese, Beijing 2, Shanghai 3, Macao 4, Tokyo 5, Japan 6. So, I now know my variables. So, I have write it is this bit came out to be 5 plus 1 divide by number of words in class Chinese add for 8 vocabulary size was 6, this comes out to be 6 by 14, what about this? So, number of times Chinese occur in class Japanese was once plus 1, number of words in class Japanese were 3 plus 6. So, this comes out to be 2 by 9.

Similarly, now you will compute the other 2 probabilities that is probability Tokyo given c, Japan given c and Tokyo given j, Japan given j. So, let us see these on slides. So, once you know how to do that we can see that quickly on the slide. So, first you complete the priors that you know are 3 by 4, 1 by 4 then you complete all these condition probabilities. So, we already completed probability Chinese given c and Chinese given j, let us see the other 2. Tokyo given c would be number of times Tokyo occurs in class c 0 plus 1, 1 divide by 14. So, denominator will remain the same.

Tokyo given j will be 1 plus 1 2 divide by 9. So, in the Japan given c will be 1 by 14, Japan given j will be 2 by 9 and that is what you see 3 by 7, 1 by 14, 1 by 14, 2 by 9; 2 by 9, 2 by 9. Now can you compute the probability of class c given document 5? Probability of class c given document 5 that again comes out to be the same formula that we wrote and we have now all these values see you put all these values. So, this will be 3 by 4 times 3 by 7 to the power 3 times 1 by 14 times 1 by 14 and this will be 1 by 4 times 2 by 9 to the power 3 times 2 by 9 times 2 by 9.

And that will give you class c has a higher probability than class j. So, then this document d 5 will be assigned to class c and not to class j and this is how you use naive Bayes model its very simple to implement and you just need to compute the probabilities of different words given different classes and use that at 1 is smoothing.

(Refer Slide Time: 07:32)

*In general, NB classifier can use any feature*  
URL, email addresses, dictionaries, network features

*But if we use only the word features and all the words in the text*  
Naïve Bayes has an important similarity to language modeling.  
Each class can be thought of as a separate unigram language model.

Pawan Goyal (IIT Kharagpur)      Text Classification - II      Week 11, Lecture 5      8 / 15

We saw, how do we use naive Bayes model of a classification now? So, there is one thing that I wanted to discuss is that how this is very close to the language model topic that we had discussed. So, how you can think of naive Bayes model as some sort of language model also. So, in general when you talk about naive Bayes model it is generic enough that can you (Refer Time: 07:58). So, you can use so if (Refer Time: 08:00) you can use what are the URLs, what are the email addresses, you can use some dictionaries, whether this word occurs in 1 of these dictionaries, what are the adverse features? For example, how many times you are getting emails from this person and so on.

Here you are allowed to use all these features inside a naive Bayes model, but suppose you are using only the content feature that is, what are all the words that are occurring in this text? So, if you use only the word features and all the words in the text then it has a very important similarity to the language modeling and what is that similarity if you think about it? So, it is like as if for different classes, you are having different language models. So, suppose a positive plus a negative plus and this is as if you are building language models for each of the class and then when you have a document at the test time, you are finding out which of this language model assigns a higher probability to this new document and that is the class of the document. So, this is an important similarity of language modeling and naive Bayes. So, you can think of each class as a separate unigram language model see; let us see some example.

(Refer Slide Time: 09:17)

*Naïve Bayes as Language Modeling*

Which class assigns a higher probability to the sentence?

Model pos		Model neg	
0.1	I	0.2	I
0.1	love	0.001	love
0.01	this	0.01	this
0.05	fun	0.005	fun
0.1	film	0.1	film

	I	love	this	fun	film
I	0.1 0.2	0.1 0.001	0.01 0.01	0.05 0.005	0.1 0.1

$P(s|pos) > P(s|neg)$



Pawan Goyal (IIT Kharagpur)      Text Classification - II      Week 11, Lecture 1

Like suppose you have 2 classes; positive and negative, now from training data you know what are the sentences that have labeled positive, you know sentences that have the label negative. So, take all the documents with the label positive edge 1 corpus and construct a language model out of that. Call it to a language model for the positive plus, similarly take all the words from the or all the documents from negative class, take it to a new corpus and build a language model unigram model, call it your negative model and now you find out the probability, you have the probability of different words as per these classes, suppose like your positive model gives a probability of 0.1 to I, 0.1 to love, 0.01 to this, 0.05 to fun and 0.1 to film. So, that is a in positive plus, you have a lot of times words like love, fun, etcetera coming. On the other hand, negative model you will not have these terms quite often. So, you will have I is occurring much more times, but love is occurring like point with a probability 0.001, fun with 0.005 and so on.

Now what will happen? At test time when you see a new document, so when you see a new document you will have a (Refer time: 10:36) you know all the words. Now try to assign this a probability as per both of these models. So, what is the probability for this sentence, I love this fun film as per my positive model and what is the probability for the sentence as per my negative model? And then you will see that immediately so positive model gives me a much higher probability than negative model and you can assign it to the positive plus and you can see that this is very much resembling, what you did in the naive Bayes model except that in naive Bayes model, you are also assigning a prior probability to each of these classes. So, here if both classes; they are roughly coming equal number of times in the data then this is like they

are very much similar, either use language model or naive Bayes.

This is another way of thinking about this problem. So, think as if you can construct different language models for each of the classes and try to assign probabilities as per different language models. Now so, we talked about the case where there are 2 classes of multiple classes and a test document can belong to one of these classes only. So, you will use a naive Bayes model or any other model and find out what is the class that could be assigned to this test document.

(Refer Slide Time: 11:56)

The slide has a blue header bar with the title "Naïve Bayes: More than Two Classes". Below the header, there is a diagram illustrating a "Multi-value classification" system. The diagram shows a central box labeled "A document can belong to 0, 1 or > 1 classes". Above this box is a section titled "Handling Multi-value classification" containing two bullet points:

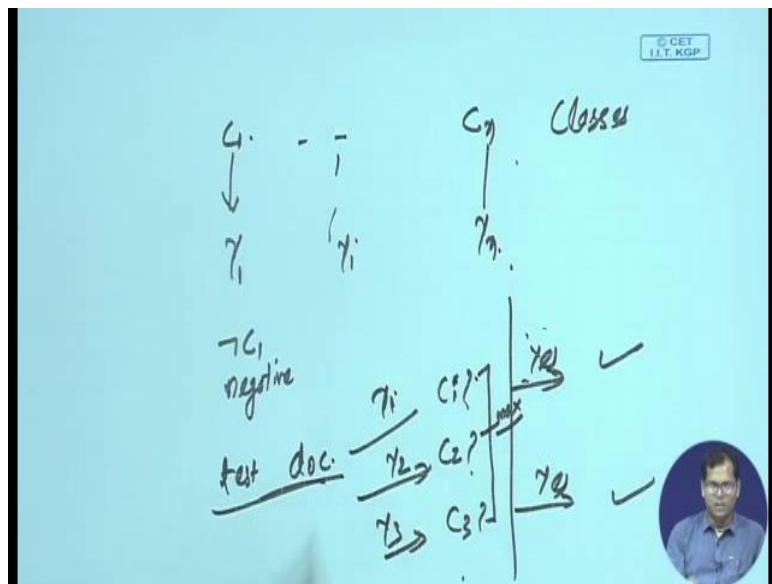
- For each class  $c \in C$ , build a classifier  $\gamma_c$  to distinguish  $c$  from all other classes  $c' \in C$
- Given test-doc  $d$ , evaluate it for membership in each class using each  $\gamma_c$

At the bottom of the slide, there is a footer bar with the names "Piyan Goyal (IIT Kharagpur)", "Text Classification - II", and "Week 11, Lect 1". To the right of the footer, there is a circular profile picture of a man.

But suppose we are talking about a multi value classification problem, so what is a multi value classification problem? A document need not be assigned to 1 and only 1 class. So, it might happen that the document does not belong to any of the classes. So, it belongs to 0 classes in the set, it might have belongs to 1 class or it might belong to more than 1 classes also. So, that is where your categories are not mutually exclusive. So, your document can belong to many of the categories at the same time.

Then how do you solve this problem using whatever we discussed, whatever technique we discussed. And a simple approach would be you make different binary classifiers for each of the classes. So, you have capital C classes, you have different binary classifiers for each of the classes and given a test document you try to assign a probability of this document to belong to each of the classes separately.

(Refer Slide Time: 12:56)



What we are saying? Suppose you have  $C_1$  to  $C_n$ ;  $n$  classes so, what you will do? So you will take 1 class and build a binary classifier  $\gamma_1$ , the document belongs to this class or does not belong to this class,  $\gamma_i$   $\gamma_n$  and different classifiers. How will you win this classifier? You need the data. So,  $C_1$  will be the positive examples, what would be the negative examples? Everything that is not  $C_1$ , here the negative examples, so you can take any of these classes. So, similarly you can build all of these classifiers. Now once you build this classifier given a document a test time, test document; you whether know all these classifiers and find out whether it belongs to class  $C_1$   $C_2$   $C_3$  and so on and whatever the classifiers says, yes, is the label given to the document. So, it might say yes in for 2 classes and it gets 2 labels.

The classifier might say no for each of the classes then it will not get any label. So, this in general, solves the problem of multi value classification where each document attach time might be given multiple different labels. So, what are we doing? So, for each class in my set, building a classifier  $\gamma_C$  that distinguishes this class  $C$  from all of the classes. Then given test document  $d$  you are evaluating its membership in each of these classes by the classifier  $\gamma_C$  and wherever  $\gamma_C$  returns to that is a label belonging to the document. So, document by this way can get many of the labels. Now this is a multi value classification.

(Refer Slide Time: 14:53)

*Naïve Bayes: More than Two Classes*

*One-of or multinomial classification*  
Classes are mutually exclusive: each document in exactly one class

*Binary classifiers may also be used*

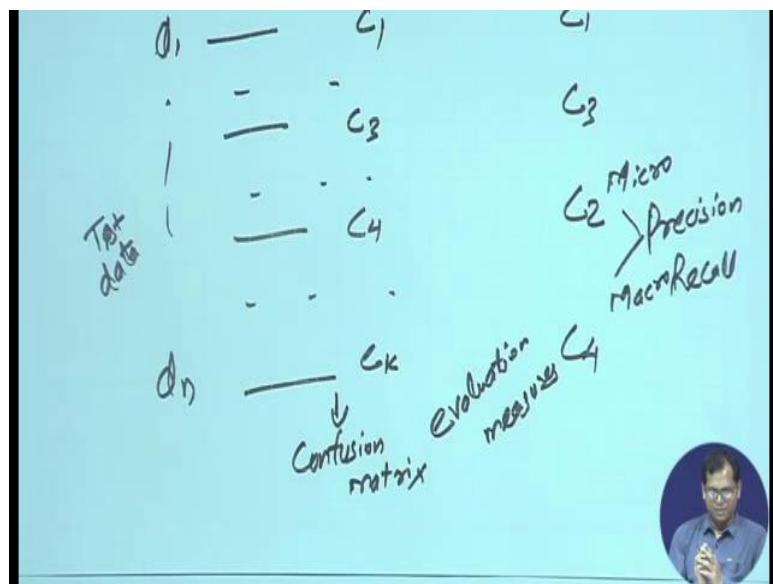
- For each class  $c \in C$ , build a classifier  $\gamma_c$  to distinguish  $c$  from all other classes  $c' \in C$
- Given test-doc  $d$ , evaluate it for membership in each class using each  $\gamma_c$

Pawan Goyal (IIT Kharagpur)    Text Classification - II    Week 11, Lec 1

On the other hand you might have this kind of a mutually exclusive classification where each document can belong to 1 and only 1 class. So, we can always handle this problem by defining multiple classes from a naive Bayes model, but suppose I am trying to build binary classifiers. So how do I handle this with binary classifiers? So, what you will do here? For each classifiers  $C$ , for each class  $C$ , you build a separate classifier like the way we did here. You build separate classifier for each of the classes. So, till here everything is same. So, each classes you are having separate classifiers test document you again run various classifiers like gamma 1 tell me whether  $C_1$  is the; so now, instead of just saying yes and no, what you will say? You will say, what is the probability that this classifier gamma 1 is giving to class  $C_1$ ?

What is the probability that gamma 2 is giving to  $C_2$ , what is the probability that gamma 3 is giving to  $C_3$ ? We will get all these probabilities and whichever probabilities, the maximum you will assign that class to the test document if the class is are mutually exclusive. So, if they are not, simply take a decision when each of the cases if they are mutually exclusive, you will find a probabilities and take the max. So, that is another approach. So, given test document  $d$  you evaluate membership for each of the classes and  $d$  will belong to the class with the maximum score. So, that is how you can build multi class classification by using simple binary classifiers. So, now, let us talk about the evaluation part. How do we evaluate our text classification approach? So, in general assume that you have some  $n$  number of classes and so what do I mean by evaluation? So, you will have a training data to train your classifier, but there will be a test data where you will find out how good your classifier is predicting the classes.

(Refer Slide Time: 17:14)



Attach data, what will happen? You will have some documents. So,  $d_1$  to some  $d_n$  and document into your test data, now you will run your classifier for each of these  $n$  documents and you will predict a class, suppose you say this belongs to class  $C_1$ , this belongs to class  $C_3$ , this belongs to class  $C_4$ , this belongs to class  $C_k$ , whatever and like that for each document, you are assigning some class separated class, now how do you know how good your model is? You have to compare it with the gold standard or the ground truth. So, then you can say that what is the true class? Suppose this is actually  $C_1$ , this is  $C_3$ , this is  $C_2$ , this is  $C_4$  and so on. This is the true class and even to match the true class with the predicted class. So, what are the different evaluation measures that can be used to compare this with this why is the simple accuracy what fraction of times your classifier gives the same answer as that ground truth that is one thing this is not very popular there are some other measures like precision recall and in precision there are 2 evaluations micro and macro.

Let us see how do we compute all these different evaluation measures? For doing that the first thing you need to do is to convert this whole thing into a confusion matrix. So, what is the confusion matrix in confusion matrix? So, what do you will have for each class how many labels were there in the predicted in the true class and how many you could predict. So, it will be like a. So, it will be what how many you got it correct how many you not get it correct everything will be there in that confusion matrix.

(Refer Slide Time: 19:26)

*Evaluation: Constructing Confusion matrix c*

For each pair of classes  $c_1, c_2$  how many documents from  $c_1$  were incorrectly assigned to  $c_2$ ? (when  $c_2 \neq c_1$ )

Docs in test set	Assigned UK	Assigned poultry	Assigned wheat	Assigned coffee	Assigned interest	Assigned trade
True UK	95	1	13	0	1	0
True poultry	0	1	0	0	0	0
True wheat	10	90	0	1	0	0
True coffee	0	0	0	34	3	7
True interest	-	1	2	13	26	5
True trade	0	0	2	14	5	10



Pawan Goyal (IIT Kharagpur)      Text Classification - II      Week 11, Lecture 1

Let us see 1 example. So, suppose I have here 6 classes; UK, poultry, wheat, coffee, interest and trade. So, you are seeing the rows are true classes; that means how many documents in these classes and this is assigned classes. So, now, how do you read this matrix? So, what this number and 95 means that you classifier assigned 95 documents to UK and they were true, but this whole column corresponds to whatever your classifier assigned a UK. So, the assigned 10 documents to class UK while they were from class wheat. So, immediately you can see that my classifier has some confusion between the classes UK and wheat. So, that is why this is called the confusion matrix, where is my classifier confusion? So, 95 cases, it assign UK, it was actually UK 10 cases is assigned wheat, sorry UK, but it was wheat. So, let us take another 1 in poultry, what is happening? 1 case where it assign poultry, what the actual class was UK, 1 case, it has assigned poultry into actually poultry 90 cases, it assigned poultry, but it was actually wheat so; that means, the classifier is really confused in that it is assigning poultry to 90 documents that were actually wheat and similarly you can read all your confusion matrix.

Now from this confusion matrix, suppose I ask you simple question like how many documents in the test data that classifier assigned to class UK and you can simply add all these, say 95 plus 10; 105, how many did it assigned to poultry? 1 plus 2; 1 plus 1, 2, 92, 93, so on, how many documents were actually belonging to class UK? That is why you have to read the row corresponded to UK, 95, 96, 109, 110; 110 documents belong to class UK.

So, now, suppose you have this confusion matrix, you can find out where your classifier is

getting confusion all that and you can try to define your classifier with this information, but now suppose you have this final confusion matrix and you want to now see what is the precision of my classifier. So, let us see we tried to find out precision; individual precision. So, what is the precision of the class UK? By precision, I mean whatever documents my classifier assigning as UK, what fraction of them are actually UK? So, let us see, it is assigning 105 documents to UK, 95 of them are actually UK, so, precision for class UK would be 95 divided by 105.

Now, what would be the recall for the class UK? Recall means among whatever documents that are in the class UK, what fraction of the documents could my classifier accurately classify? So, here they were 110 documents in the class UK, my classifier could accurately classify 95 of those. So, recall would be 95 divided by 110, like that you can compute for each of the classes the individual precision and recall.

(Refer Slide Time: 22:47)

*Per class evaluation measures*

**Recall**  
Fraction of docs in class  $i$  classified correctly:  $\frac{c_{ii}}{\sum_j c_{ij}}$

**Precision**  
Fraction of docs assigned class  $i$  that are actually about class  $i$ :  $\frac{c_{ii}}{\sum_i c_{ji}}$

**Accuracy**  
Fraction of docs classified correctly:  $\frac{\sum_i c_{ii}}{N}$

Piwan Goyal (IIT Kharagpur)    Text Classification - II    Week 11, Lecture 5    13 / 15

Recall would be whatever fraction of documents in class  $i$  corrected classified correctly and that you can see you can get by adding overall the all the elements in the row like that is what we did for the class UK added all the elements in the row  $C_{ii}$  divided by summation over  $C_{ij}$   $c_{ij}/\sum_j c_{ij}$  for all  $j$  and precision fraction of documents that are assigned to class  $i$  that are actually about class  $a$ . So, this will be again  $C_{ii}$  divided by now you will add the whole call.  $c_{ii}/\sum_j c_{ij}$  So, summation over  $i$   $C_{ji}$  and accuracy would be add all your diagonal elements divided by the total number of test documents that would be your accuracy  $\sum_j c_{ij}/N$ .

Now, we say that we can also compute the. So, this is for each and individual class and accuracy you can you need to talk about the overall test data, but can you compute precision for. So, micro and macro average precision for this test data.

(Refer Slide Time: 23:58)

The slide has a blue header bar with the title "Micro- vs. Macro-Average". The main content area is white with a blue border. It starts with a question: "If we have more than one class, how do we combine multiple performance measures into one quantity?". Below this, there are two sections: "Macro-averaging" and "Micro-averaging".

- Macro-averaging:** Compute performance for each class, then average
- Micro-averaging:** Collect decisions for all the classes, compute contingency table, evaluate.

At the bottom of the slide, there is a navigation bar with icons for back, forward, search, and other presentation controls. The footer contains the text "Piwan Goyal (IIT Kharagpur)", "Text Classification - II", "Week 11, Lecture 5", and "14 / 15".

What is this? So, if we have more than one class then how do we combine the performance measures for individual classes to make a single evaluation measure that is my micro and macro average precision? So, what is the difference? So, macro average what I will do? I will compute the precision or accuracy precision for is in the individual class and i will taken average over each suppose my class one is having a precision of 0.7 class 2 is having a precision of 0.9 I will take an average 0.7 plus 0.9 divide by 2.8 in micro average precision what I will do? I will first come take all the decisions and compute a single matrix over all the decision over taken together. So, I will see how many cases are taking as true, false, I will take them together make a single statistic or single condition suitable and I will compute my precision recall from that.

(Refer Slide Time: 25:01)

*Micro- vs. Macro-Average*

Class 1		Class 2		Micro Ave. Table	
	Truth: yes		Truth: yes		Truth: yes
Classifier: yes	10	10	Classifier: yes	90	10
Classifier: no	10	970	Classifier: no	10	890

- Macro-averaged precision:  $(0.5 + 0.9)/2 = 0.7$
- Micro-averaged precision:  $100/120 = 0.83$

Micro-averaged score is dominated by score on common classes



Pawan Goyal (IIT Kharagpur)      Text Classification - II      Week 11, Lec 1

Let us take an example suppose for the class one there were 10 cases where you classifier said yes and the actual answer was also yes 10 cases where classifier said yes, but they were not belonging to the class 1, 10 cases where your classifier said no and they were actually belonging to the class one and are the 970 cases where classifier said no and they were not belonging to class 1 and this is one of the way of writing the confusion matrix for contingency suitable for each class. So, that is classifier yes no; truth yes no.

From there can you compute, what is the precision for class 1? So, it classify assign 20 documents to class one out of them 10 were correct precision is 0.5. Now let us see the second class, in the same manner we constructed the contingency table. So, what is the precision for class 290 were assigned directly out of 100 given by the classifier. So, it is 0.9.

So, now, when we have to do micro average precision we combine all these statistics. So, we combine these statistics say they were 10 plus 90, 100 cases where classifier said yes and this was also yes 10 plus 10; 20 cases were classifier said yes, but it was not the correct class similarly 20 cases here 160 cases here. So, this is you combine all these statistics and do a single table that becomes your micro average table. Now from once you have this table, how do you compute your macro average precision and macro average precision. So, micro average precision you compute precision for class 1, class 2 taken average. So, you have already computed this is 0.5 and this is 0.9 taken average. So, it becomes 0.7. So, this is your macro average precision.

Now, what to do a macro average precision? So, you will compute precision over this table now. So, that will be 100 divided by 120. So, that is point roughly 0.83. So, micro average precision comes out to be 0.7 and micro average comes out to be 0.83. So, now, what do you see o. So, why micro is coming out to be higher than macro precision? So, because in macro precision, you are giving in equal weight to all the classes, you are computing precision over class 1, classes 2 then you forget how many instances where there for each of the class. So, and they are given equal weight and you compute an average precision in micro average precision what is happening the class that is having more number of instances getting a higher weightage that is why it is bias towards class with classes that are having higher number of instances.

If you act if you are building a classifier of a multiple classes and you want to see that all your class if classes perform well then macro average precision is a good precision, if your classes are imbalanced, some classes are having high number of data points and other. So, macro average score is dominated by the score on the common classes. So, when you are talking dealing with test classification they might be some issues like in your training data some classes are more common than others. So, here more samples from some classes than other classes and this sometimes Bayes your classifier also. So, good strategy is that your sample, the number of instances in the classes.

So, one simple thing is you can do under sampling. So, if a classes very common you under sample it such that it becomes close to the other classes that are quite rare. So, that by that way in you are training data you will have roughly equal amount of samples in each of the classes and there are other algorithms that also allow you to over sample some of the minority class and so on. So, we will not discuss that in detail just wanted to tell that it might happen that you have an unbalanced data set then you have to do certain strategies before applying the classifier.

So, that finishes our week eleven for this course and then. So, next week we will discuss in detail about the sentiment analysis and opinion in next lecture. So, this will be topic for the final week.

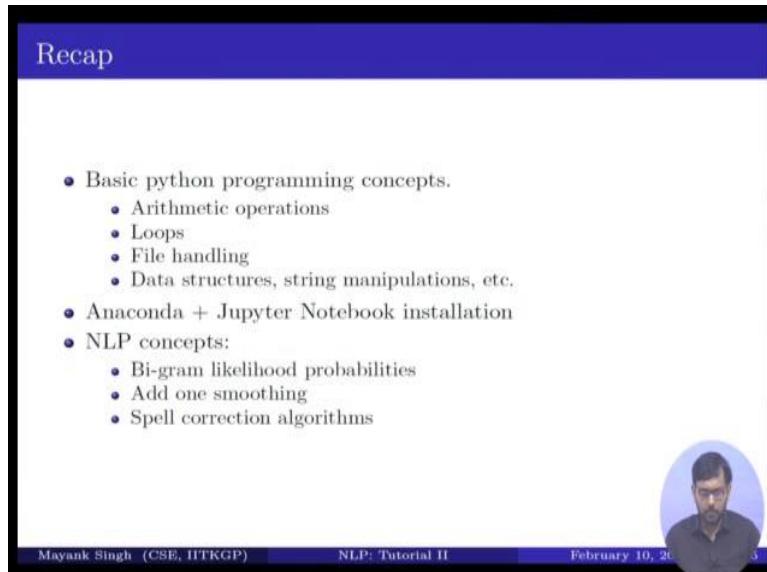
Thank you.

**Natural Language Processing**  
Prof. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 57**  
**Tutorial**

Hello everyone, my name is Mayank Singh. So, today I will be going through the second tutorial for this course.

(Refer Slide Time: 00:25)



Recap

- Basic python programming concepts.
  - Arithmetic operations
  - Loops
  - File handling
  - Data structures, string manipulations, etc.
- Anaconda + Jupyter Notebook installation
- NLP concepts:
  - Bi-gram likelihood probabilities
  - Add one smoothing
  - Spell correction algorithms

Mayank Singh (CSE, IITKGP)      NLP: Tutorial II      February 10, 2024



In the previous tutorial we have gone through the basic python programming concepts for example arithmetic operations, loops, file handling, data structures, string manipulations, etcetera. We have also shown steps to install anaconda and Jupyter notebook installations along with that we have shown some NLP concepts for example, how to compute bigram likelihood probabilities, how to do add one smoothing and how to correct spellings using 2 famous spell correction algorithms.

(Refer Slide Time: 00:53)

What Next ...

- Introduction to NLTK.
- Analyze large Corpus
- Visualize empirical laws
- Brief introduction to POS tagging

Mayank Singh (CSE, IITKGP) NLP: Tutorial II February 10, 2024



What are we going to do in this tutorial? In this tutorial, I will start with the brief introduction to NLTK. NLTK is a python toolkit for natural language and processing tasks. We will use NLTK to analyze large corpus and how to how we can visualize some empirical loss. Towards the end I will also give you a brief introduction to pos tagging.

(Refer Slide Time: 01:19)

Prerequisite

- Install **NLTK** from Anaconda navigator. Then download:
  - **Punkt Tokenizer** (Models tab)
  - **Averaged Perceptron Tagger** (Models tab)
  - **Brown corpus** (Corpora tab)

<https://github.com/mayank4490/NLP-tutorial-2>

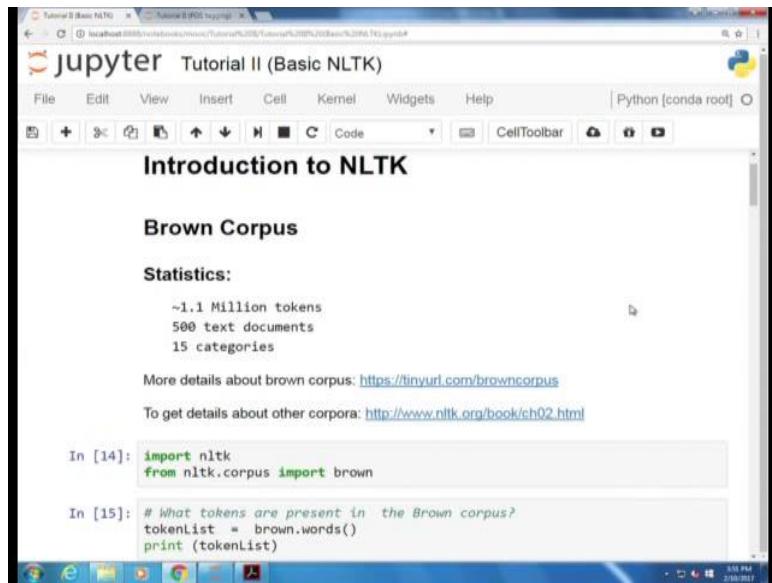
Mayank Singh (CSE, IITKGP) NLP: Tutorial II February 10, 2024



But before you start programming for this tutorial, you have to install NLTK from anaconda navigator. Once NLTK is installed, you have to install and download Punkt Tokenizer and averaged Perceptron tagger from the model step and brown corpus from the corpora tab. For

more information about the NLTK and the installation part, you can click on this link.

(Refer Slide Time: 01:57)



The screenshot shows a Jupyter Notebook interface with the title "jupyter Tutorial II (Basic NLTK)". The notebook displays an introduction to the Brown Corpus, which is described as having approximately 1.1 million tokens from 500 text documents, categorized into 15 categories. Below this, two code cells are shown:

```
In [14]: import nltk  
from nltk.corpus import brown  
  
In [15]: # What tokens are present in the Brown corpus?  
tokenList = brown.words()  
print(tokenList)
```

I assume that you can successfully install all these dependencies. So, let us move to the coding part, yes. So, this tutorial is divided into 2 python note books, in the first note book I will give you a brief introduction to NLTK and how we can process large corpus using NLTK, as an intuitive use case we have used brown corpus. So, in brown corpus is a very old English corpus that consists of 1.1 million tokens from 500 text documents all these text documents are divided into 15 categories. For more information about brown corpus you can visit this link.

(Refer Slide Time: 02:35)

```
In [27]: import nltk  
from nltk.corpus import brown  
  
In [15]: # What tokens are present in the Brown corpus?  
tokenList = brown.words()  
print (tokenList)  
  
[u'The', u'Fulton', u'County', u'Grand', u'Jury', ...]  
  
In [16]: # What is the size of Brown corpus?  
print ("Total token count:", len(tokenList))  
  
('Total token count:', 1161192)  
  
In [17]: # What are the names of the 15 categories?  
print (brown.categories())  
  
[u'adventure', u'belles_lettres', u'editorial', u'fiction', u'government', u'hobbies', u'humor', u'learned', u'lore', u'mystery', u'news', u'religion', u'reviews', u'romance', u'science_fiction']
```

There are many other corpora in different languages in NLTK. So, you can get details about those corpora also on the second link. So, let us first set up our environment. So, first of all, we need to import NLTK. So, importing NLTK means that we are going to load all the dependencies of NLTK using import NLTK. In the second line we are importing brown corpus using NLTK corpus. So, let us run this, yes. So, now, our initial set up is done our libraries and the brown corpus is loaded into the python note book.

Now, first question is what tokens are present in the brown corpus? So, what does it mean? So, brown corpus is a very large corpus it consists of 1.1 million tokens, but what are those tokens. So, brown dot words function will give you the list of the words or tokens that are present in the brown corpus. What I am doing here is I am storing all these tokens into a list called token list and then printing out that list. So, if I am doing this, I am getting the output as the Fulton, County, Grand, Jury, etcetera that is these are the first 5 first 6 tokens from the brown corpus.

Now, the next question is what is the size of the brown corpus? To get the size of the brown corpus, we just need the length of the token list. So, I am using the python length function to get the length of the brown corpus. So, what I am doing is length token list will print out the length of the token list and that is also equal to the size of the brown corpus. So, if I run this I will get that token total token count is around 1.1 million tokens. So, as I have already mentioned that the brown corpus is categorized into 15 different categories. So, what are the

names of those categories? So, if I run the function brown dot categories, it will list the names of different categories that are present in the brown corpus.

For example, if I run this, it gives the category names as adventure, belles, lettres, editorial, fiction, government, hobbies, humor, learned, lore, mystery, news, religion, reviews, romance, science, fiction. So, similar as before, we can ask a very similar question that what are the tokens that are present in our adventure category that is given a particular category what are the tokens that are present?

(Refer Slide Time: 05:30)

The screenshot shows a Jupyter Notebook interface with the title "Tutorial II (Basic NLTK)". The notebook contains several code cells and their corresponding outputs:

- In [18]:

```
# What tokens are present in "Adventure" category?
adventureTokenList = brown.words(categories='adventure')
print(adventureTokenList)
[u'Dan', u'Morgan', u'told', u'himself', u'he', ...]
```
- In [19]:

```
# What is the size of "Adventure" category?
print("Token count of adventure category:", len(adventureTokenList))
('Token count of adventure category:', 69342)
```
- In [20]:

```
# How many unique tokens are present in Brown corpus?
print("Brown vocabulary size:", len(set(tokenList))) # Also token types
('Brown vocabulary size:', 56057)
```
- In [21]:

```
# Can we compute TTR now?
print("TTR for Brown corpus:", len(set(tokenList))/ float(len(tokenList)))
('TTR for Brown corpus:', 0.048275392872152066)
```
- In [22]:

```
# How many times each token appears in the corpus?
```

Again we will use the same function as before that is brown dot words, but this time the parameter for this function will be the name of the category.

For example, for adventure category the parameter will be categories is equal to adventure. So, the output will be a list of tokens and we are storing that list in a variable called adventure token list then we print the adventure token list variable. So, you can see the tokens are Dan, Morgan, told, himself, he and so on and so forth. Now what is the size of the adventure category? This is again similar to the size of the entire brown corpus.

What I am going to do is I will again find the length of the list of adventure tokens. So, this is what I have done here I have just use the len function on the parameter that is adventure token list, please see that here if you see the names here like in 18 cell, the names are the tokens actually are Dan Morgan which is a name then, told, himself, he, all these tokens are

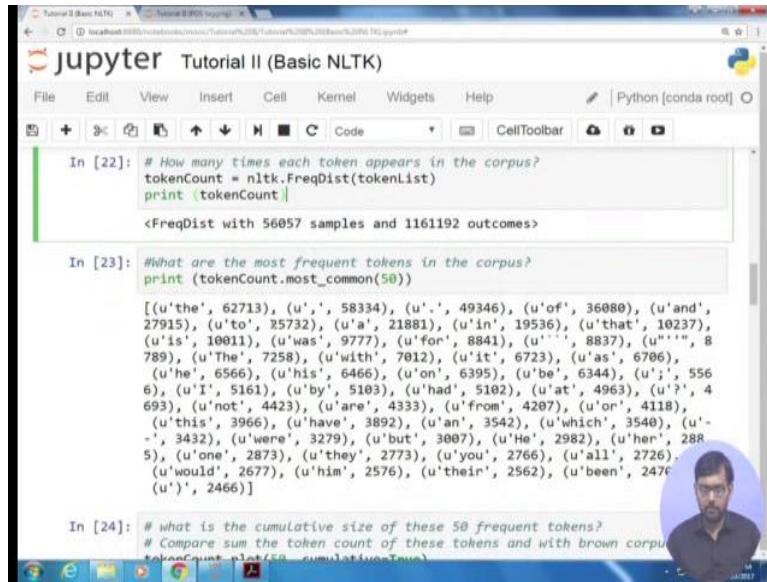
preceded by a character u. So, u here means that all these tokens are actually Unicode strings; these are not ASCII strings fine. So, till now, we have done, we have got the list of the tokens that are present in the brown corpus, we have got the size of the brown corpus, we also know that the different categories of the brown corpus then we can know the different words that are present in the brown; different categories of the brown corpus and finally, the length of or the size of the each category.

Now, the question is how many unique tokens are present in the brown corpus? So, for this we are using the set function that is a part of python library. So, what is set function? Set function will actually convert a list into a set as you know that in a set all these elements are unique. So, what I am doing here is first of all I will type cast that token list into a set. So, set will have the unique elements and then I will find the length of that set fine. So, if I run this, I will get the set sizes around 56000 that is the brown vocabularies size is 56000. So, these are the unique number of tokens that are present in the brown corpus.

Please note that here I have not converted any of the tokens into lower case. So, please if you want to convert these tokens into lower case then the total vocabulary size will be even lesser. Now can we compute TTR? Now if you remember the TTR was discussed in the first week of the lectures. So, TTR is a ratio of the number of unique elements upon total number of elements. So, here we have just computed the number of unique elements or the tokens and we have computed here the number of total number of tokens.

Let us compute TTR. So, the TTR value here is 0.0482 which means that on an average each token is 21 times present in the corpus. So, till now, we have not done any token wise analysis so; that means, we do not know how many times each token appears in the corpus.

(Refer Slide Time: 09:27)



The screenshot shows a Jupyter Notebook interface with three code cells. Cell [22] contains code to calculate the frequency distribution of tokens in a corpus. Cell [23] prints the 50 most common tokens. Cell [24] calculates the cumulative size of these tokens. A circular profile picture of a man with glasses is visible in the bottom right corner of the slide.

```
In [22]: # How many times each token appears in the corpus?
tokenCount = nltk.FreqDist(tokenList)
print(tokenCount)

<FreqDist with 56057 samples and 1161192 outcomes>

In [23]: # what are the most frequent tokens in the corpus?
print(tokenCount.most_common(50))

[(u'the', 62713), (u'.', 58334), (u'.', 49346), (u'of', 36080), (u'and', 27915), (u'to', 25732), (u'a', 21881), (u'in', 19536), (u'that', 18237), (u'is', 10811), (u'was', 9777), (u'for', 8841), (u'', 8837), (u'', 8789), (u'The', 7258), (u'with', 7012), (u'it', 6723), (u'as', 6706), (u'he', 6566), (u'his', 6466), (u'on', 6395), (u'be', 6344), (u';', 5566), (u'I', 5161), (u'by', 5103), (u'had', 5102), (u'at', 4963), (u'?', 4693), (u'not', 4423), (u'are', 4333), (u'from', 4207), (u'or', 4118), (u'this', 3966), (u'have', 3892), (u'an', 3542), (u'which', 3540), (u'-', 3432), (u'were', 3279), (u'but', 3007), (u'He', 2982), (u'her', 2885), (u'one', 2873), (u'they', 2773), (u'you', 2766), (u'all', 2726), (u'would', 2677), (u'him', 2576), (u'their', 2562), (u'veen', 2476), (u')', 2466)]

In [24]: # what is the cumulative size of these 50 frequent tokens?
# Compare sum the token count of these tokens and with brown corpus
tokenCount.ploth(50, cumulative=True)
```

It may appear in the corpus or we may talk about the category also like how many times each token appears in a particular category, to get that information we have used here the NLTK function frequency distribution and the parameter for this function is that token list. So, token list if you remember is actually the list of the tokens that are present in the brown corpus.

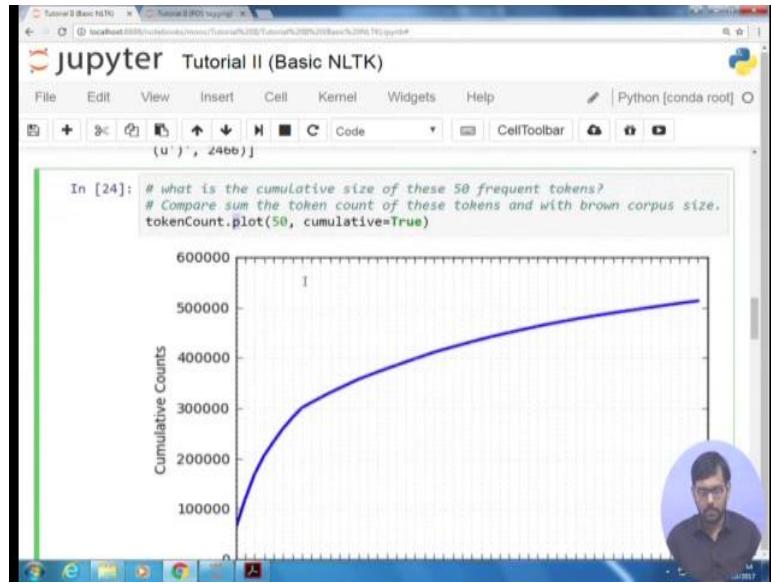
What I am doing here is I am passing the list of the tokens that are present in the brown corpus to the frequency distribution function that is a part of NLTK. So, these functions written that dictionary of tokens with their counts so, the dictionary token count will have the key as the token name and the value will be how many time that token occurs in the corpus. So, now, we have the information stored in the token count that how many times a particular token has occur in the corpus. So, can we get the most frequent tokens in the corpus? Yes, we can use the most common function that is a part of the NLTK.

What we are going to do if we write token count dot most common 15? So, it wins out 50 most common tokens from the token count dictionary. So, as you can see here it is it has printed the list of first most frequent 50 tokens. So, as you can see the has been the most frequent token then by then comma then full stop and of and to a in that etcetera. So, most of the words here are almost all the words here are the function words. So, this is as for our knowledge that functional words are the majority of the words in any of the corpus.

Now, what we wanted is, what is the cumulative size of these 50 frequent tokens? That is we want to compare the sum of the token count of these 50 frequent tokens with the entire brown

corpus size. So, for that I have used a plot function.

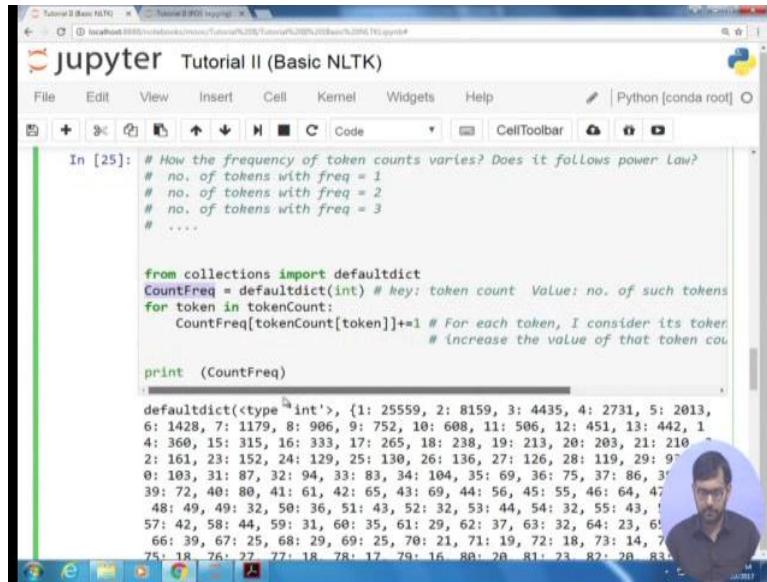
(Refer Slide Time: 11:51)



What I am plotting here is I am plotting the token count of 50 most frequent tokens and cumulative wise that is on the x axis we have the tokens and on the y axis we have the cumulative token counts. So, we start from that and then from on onwards we add the previous token count and we get the cumulative value.

The main observation here is that the size of these 50 tokens is almost half of the size of the brown corpus that is these size; these tokens cover almost half of the brown corpus finally, in the end to in the end of the analysis part, how we ask very important question that is how the frequency of token count varies and does it follows power law that is what we wanted to ask is what is the number of tokens with frequency 1? What is the number of tokens with frequency 2 and so on?

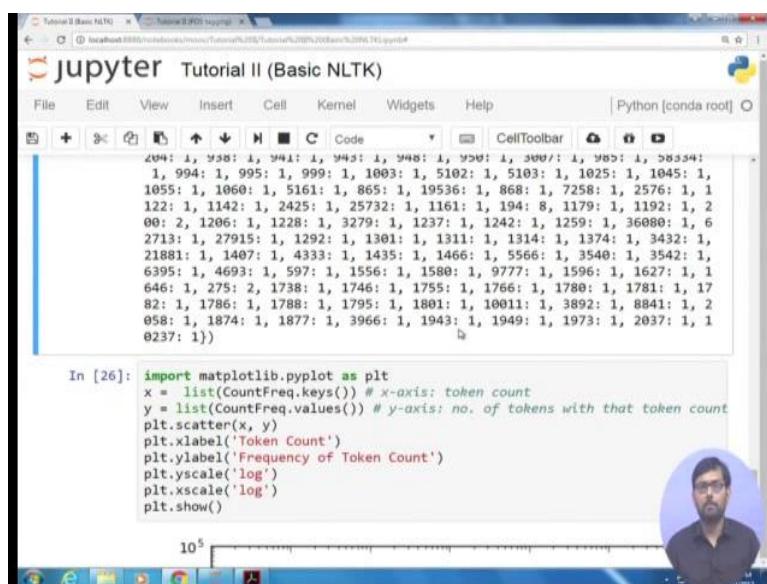
(Refer Slide Time: 12:59)



```
In [25]: # How the frequency of token counts varies? Does it follows power Law?  
# no. of tokens with freq = 1  
# no. of tokens with freq = 2  
# no. of tokens with freq = 3  
# ....  
  
from collections import defaultdict  
CountFreq = defaultdict(int) # key: token count Value: no. of such tokens  
for token in tokenCount:  
    CountFreq[tokenCount[token]]+=1 # For each token, I consider its token count  
    # increase the value of that token count  
  
print(CountFreq)  
  
defaultdict(<type 'int'>, {1: 25559, 2: 8159, 3: 4435, 4: 2731, 5: 2013,  
6: 1428, 7: 1179, 8: 986, 9: 752, 10: 608, 11: 506, 12: 451, 13: 442, 1  
4: 360, 15: 315, 16: 333, 17: 265, 18: 238, 19: 213, 20: 203, 21: 210  
2: 161, 23: 152, 24: 129, 25: 130, 26: 136, 27: 126, 28: 119, 29: 97  
0: 103, 31: 87, 32: 94, 33: 83, 34: 104, 35: 69, 36: 75, 37: 86, 38:  
39: 72, 40: 80, 41: 61, 42: 65, 43: 69, 44: 56, 45: 55, 46: 64, 47:  
48: 49, 49: 32, 50: 36, 51: 43, 52: 32, 53: 44, 54: 32, 55: 43, 56:  
57: 42, 58: 44, 59: 31, 60: 35, 61: 29, 62: 37, 63: 32, 64: 23, 65:  
66: 39, 67: 25, 68: 29, 69: 25, 70: 21, 71: 19, 72: 18, 73: 14, 74:  
75: 18, 76: 27, 77: 18, 78: 17, 79: 16, 80: 20, 81: 23, 82: 20, 83: 18}
```

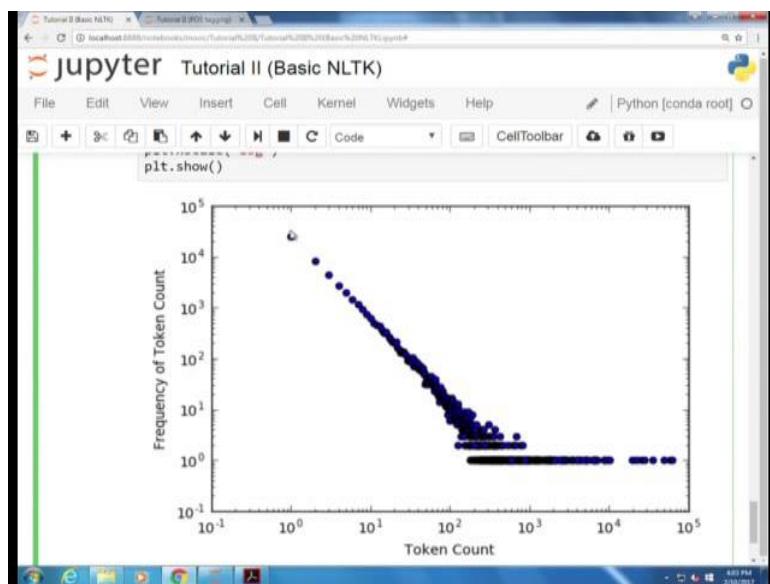
For this, we initialize a dictionary called Countfreq. So, Countfreq will store the frequency of the token counts that is the key will be the token count and the value will be the number of such tokens with that particular token count we iterate over the token count dictionary to populate count frequency dictionary. So, what we are going to do is for each token in count token count I will consider it is token count and increase the value of that token count in count frequency dictionary. So, this is what I have done here and then I have printed count frequency dictionary. So, as you can see here it prints large number of token counts along with their frequency.

(Refer Slide Time: 13:37)



What it says is there exists around 25000 tokens which have just 1 frequency, similarly there exists around 8159 tokens which occur twice in the corp brown corpus and so on, now can we plot this distribution? Yes, we can plot this distribution using Matplotlib. Matplotlib is a very useful tool for plotting in python.

(Refer Slide Time: 14:17)



What I am doing here is on x axis? I have stored the token counts and on the y axis, I am storing the number of tokens with that particular token count then I am just plotting a scatter plot x comma y and this plot for this plot, we have scaled x and y axis on the log scale. So, as you can see, this is a very famous power law plot and this gives the t log that power log. So, on x axis we have the token count on y axis we have the frequency of the token count. So, you can see. So, this actually concludes our first part of the tutorial that is the analysis of corpus using a NLTK.

For more information, you can question on the you can click on the link that I have provided in the slide and you can for any installation query also you can question on that same name now going to the second part of the tutorial that is a basic introduction to pos tagging. So, first of all what am doing here is I have imported a NLTK. So, let us first import NLTK that is the basic requirement for the second tutorial. This NLTK is now given a sentence can we pos tagging. So, the sentence here is this is a basic tutorial on pos tagging.

First of all, we will tokenize this sentence and then we will pass it to the pos tagger. So, what we are going to do here is NLTK dot word tokenize sentence. So, word tokenize is the

function of NLTK that will tokenize the sentence and the tokenize sentence will be pass to the pos tagger. So, the tokenize sentence is stored in a Tokenizedsent variable and we have print the Tokenizedsent variable here. So, Tokenizedsent consist of nine tokens.

Now we have then we have directly pass Tokenizedsent into the pos tag function which is actually a pos tagger of NLTK and we have printed the output. So, the pos tagger has pos tag the Tokenizedsent list. So, let us see the result. So, this here is the determiner is here is third person singular present, a is again determiner, basic is adjective, tutorial is singular now, on is preposition is proper noun singular tagging is singular noun and then we have the period. So, here we have given a example sentence, but we can also do same for the brown corpus.

First of all, let us import the brown corpus. So, we have imported brown corpus as we have done before say from NLTK dot corpus import brown now what are we going to do next is how to get sentences from the brown corpus. So, brown dot sentence actually gives you tokenized sentences from brown corpus. So, if I write brown dot sent and we print the result we get the sentences. So, the first sentence is the Fulton county grand jury said Friday an investigation of Atlanta's recent primary election produced no evidence that any irregularities took place and then sentence ends.

Similarly, we have the other sentence and the sentence (Refer Time: 17:50). So, we directly pass each of the sentence into the pos tag function and we get the pos tagged output. So, what we are doing what we have done here is we have passed the first sentence of brown corpus into the pos tagger and we have got the result.

So, with this I would like to conclude for the tutorial and in the next tutorial we will be talking about some more advanced techniques of NLTK.

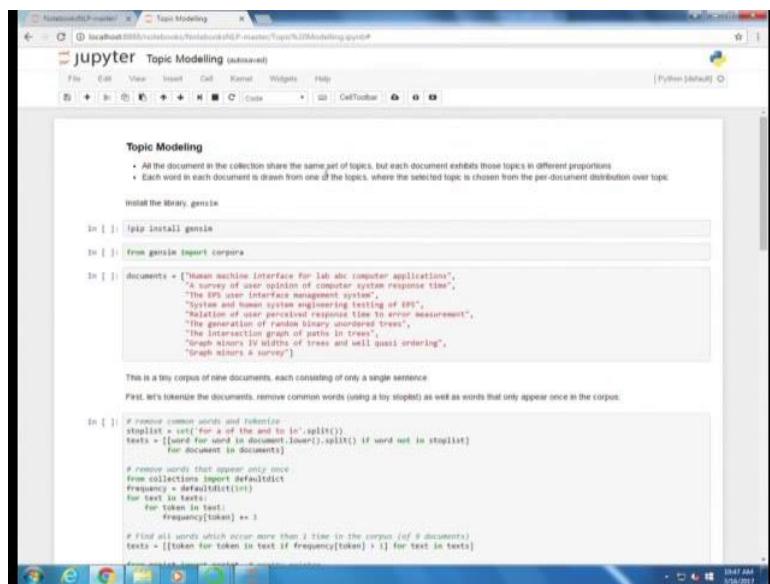
Thank you.

**Natural Language Processing**  
Prof. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 58**  
**Tutorial**

Hello everyone. So, welcome to today's tutorial session, today we will be discussing about topic modeling. So, for this tutorial, we assume that we already have Jupyter notebook installed and we will be starting with the topic modeling session.

(Refer Slide Time: 00:41)



The screenshot shows a Jupyter Notebook interface with the title "Topic Modeling". The notebook contains the following code:

```
Topic Modeling
All the documents in the collection share the same set of topics, but each document exhibits those topics in different proportions
Each word in each document is drawn from one of the topics, where the selected topic is chosen from the per-document distribution over topics

Install the library gensim
In [1]: !pip install gensim
In [2]: from gensim import corpora
In [3]: documents = ["Human machine interface for lab abc computer applications",
                 "A survey of user opinion of computer system response time",
                 "User satisfaction with computer system response time",
                 "Vortex and maven are important components of ipmt",
                 "Relation of user perceived response time to error measurement",
                 "The generation of random binary unordered trees",
                 "The generation of random binary ordered trees",
                 "Graph minors IV widths of trees and well quasi ordering",
                 "Graph minors A survey"]

This is a tiny corpus of nine documents, each consisting of only a single sentence.
First, let's tokenize the documents, remove common words (using a stoplist) as well as words that only appear once in the corpus:
In [4]: # remove common words and tokenise
stoplist = set([word for word in stop if word not in stoplist])
texts = [[word for word in document.lower().split() if word not in stoplist]
         for document in documents]

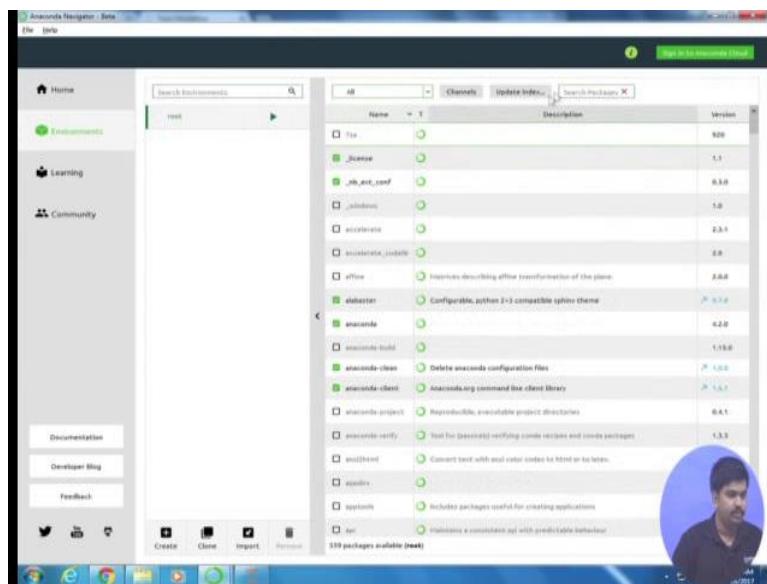
# remove words that appear only once
from collections import defaultdict
frequency = defaultdict(int)
for text in texts:
    for token in text:
        frequency[token] += 1
# Find all words which occur more than 2 times in the corpus (of 8 documents)
texts = [[token for token in text if frequency[token] >= 2] for text in texts]
```

We have started Jupyter notebook and here we will be discussing about topic modeling. So, I assume you remember, what is topic modeling? So, it essentially is a way to model a corpus or a set of documents where all the documents in the collection share the same set of topics, but each document exhibits those topics in different proportions.

We essentially assume that we have a corpus and in that corpus that contains multiple documents, every document shares a common set of topics. Now given those topics, they can be varying in different proportions in each of these documents. Now each word in the given document is basically drawn from one of those topics. So, here we assume that every word that you see in a particular document that comes from the distribution of a particular topic and based on the document wise topic proportion you get to see those words.

Now, to infer all this, all we have is basically the observations or bay or the corpus itself. So, to run this codes that we are discussing today you might be requiring to install a library called Gensim. So, I assume that everyone is familiar with anaconda navigator.

(Refer Slide Time: 02:15)



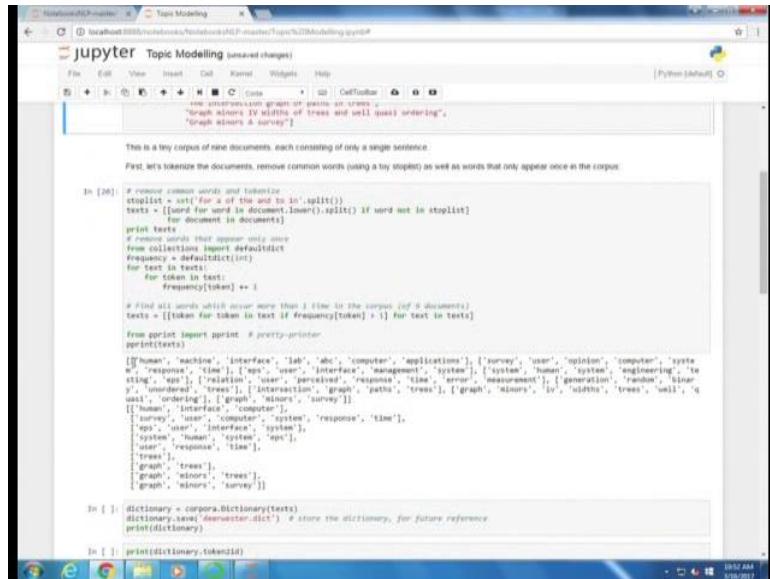
Once we use anaconda navigator have to browse to environment and look for the library section here. So, here just select on the all option and look for Gensim. So, we have already preinstalled Gensim. So, here you should ideally be not seeing the tick mark. So, select this and install the library.

Once the library is installed, we will import the library and for this tutorial, we will be assuming a very small corpus which is basically a collection of 9 different sentences. So, here is a tiny corpus of 9 documents and each document essentially contains only one sentence. So, in order to view this corpus to the model we have to do some essential pre processing. So, we will be showing what all pre processing needs to be done. So, we need to convert this to a suitable bag of words format and we will be showing how to do this.

Here are some pre processing steps that we are going to do. So, for this simplistic corpus we assume the only set of stop words are these few words, but in practicality you might have to consider other stop words as well, but there are pre defined sets that you can obtain we will show how to obtain those lists. So, we make a list of all those stop words we also make sure that all the words are converted to lowercase. So, here what we do is that we take each sentence in the documents and then we split it based on the space. So, we do tokenization

again based on a very simple notion that if there is a space between 2 characters we assume that they belong to 2 different words.

(Refer Slide Time: 04:37)



```
In [30]: # remove common words and tokens
stoplist = set([word for word in in_.split()])
texts = [[word for word in document.lower().split()] if word not in stoplist]
for document in documents:
    print text
# remove words that appear only once
from collections import defaultdict
frequency = defaultdict(int)
for text in texts:
    for token in text:
        frequency[token] += 1
# Find all words which occur more than 1 time in the corpus (of 8 documents)
texts = [[token for token in text if frequency[token] > 1] for text in texts]
from pprint import pprint
print texts
[[['human', 'machine', 'interface', 'lab', 'abc', 'computer', 'applications'], ['survey', 'user', 'opinion', 'computer', 'system'],
  ['time', 'user', 'computer', 'system'], ['user', 'perceived', 'response', 'time', 'error', 'measurement'], ['generation', 'random', 'binary',
  'unordered', 'trees'], ['interaction', 'graph', 'paths', 'trees'], ['graph', 'minors', 'by', 'widths', 'trees', 'until', 'q'],
  ['human', 'interface', 'computer'],
  ['survey', 'user', 'computer', 'system', 'response', 'time'],
  ['user', 'perceived', 'response', 'system'],
  ['system', 'human', 'system', 'sys'],
  ['user', 'response', 'time'],
  ['trees'],
  ['graph', 'trees'],
  ['graph', 'minors', 'trees'],
  ['graph', 'minors', 'survey']]
```

```
In [31]: dictionary = corpora.Dictionary(texts)
dictionary.save(deeblester.dict) # store the dictionary, for future reference
print(dictionary)
```

```
In [32]: print(dictionary.token2id)
```

Since our corpus is something that we are doing for representation purposes these simplistic method will help you in doing this, but again you can rely on any standard Tokenizers for this task. So, text will essentially contain all those individual words see if you see here the word the sentence human machine interface for lab, A B C computer applications here that document is now represented as the list of words where you can find the H in human being has been turned in this small h or the lowercase h, the stop word for is removed and we essentially got a list of list where each list shows a document with its important words.

After doing this what we find is that we find the count or the frequency of a particular word occurring in the entire corpus for example, the word human is appearing once in document one and it also appears one time in document 4. So, the total occurrence is 2 for the word human. So, this particular chunk of code basically tells you how to find the frequency of the individual words in your corpus once that is done, we filter those words which appear more than once. So, we keep those words only those words which appear more than once in our corpus and we remove them.

What you find here in the second printed statement are those words which are filtered after this particular criteria is applied. So, we can see that machine which was appearing only once in the entire corpus is removed and similarly other words like interface etcetera. So, once we

have this, most often, what happens is that when we have large corp error, we may not be doing it every time like we may not be doing the pre processing every time because it itself might be time consuming.

(Refer Slide Time: 06:42)

Once we have this pre processed state, we save it into any of the suitable formats. So, here Gensim provides a suitable format for saving this text called dictionary and we save it in that format, see if you see here in Gensim, we use the dictionary method to convert the text to a suitable internal representation and we find that there are 12 unique tokens for the entire corpus, once we have that for ease of handling the data we convert each word to a unique individual representation. So, every unique word will be given an individual ID.

There arbitrarily provided there is no rational for which number goes to which word. So, once we apply this operation dictionary dot token 2 ID which is an in built attribute for this model as given by Gensim you can find that the words like minus is given an id eleven graph is given an id ten and so on. Now we assume that whatever words are going that we are going to use will come from only one of these words.

Now, let us take a new document which is human computer interaction. So, I have a new sentence or a new document human computer interaction. So, I have to apply all those pre processing that I have applied to the corpus, since there are now stop words I do not apply the stop words filtering explicitly, but it is also assumed to be that. Now we convert it to something called as b o w or bag of words representation. So, a document is now going to be

represented in terms of the bag of words representation which is very similar to this one.

We can find that you get a list with 2 petals which show 1 comma 1 and 2 comma 1. So, what essentially is represented here? So, essentially what you can find here is that we have 3 terms human computer and interaction, but unfortunately interaction is a word that is not available in our dictionary. So, we ignore that word when we convert this document to a vectorial representation. So, for all practical purposes, you can assume your document to be human computer as this is what it is going to be fed to the topic model.

Now, what is this first entity that is 1 and 2 represent they basically represent each of the word human and computer see, if you see the identifier for human is 2 and the identifier for computer is 1 and what you see on the right is the number of times a particular word is appearing for example, if I add another term computer again. So, it is human computer interaction computer the word with ID 1 which happens to be computer as increases frequency. So, this is at basically frequency count of each unique word that basically represents the documents.

As we have saved the dictionary, we can also save these representations also in a serializable format and we can store it so that we can load it, later whenever we require that so, these are all those lines sentences that we were dealing with here, they are first converted to a list format with its relevant words. Now we convert it to the individual usage or the individual identifier along with its frequency it. So, happen that in this corpus we do not have any of the word repeating twice other than this particular word that is the 6th word in the 3rd sentence. So, word 6 is system and in 3rd sentence, yes system; human system e p s. So, that is now reduced to 6 comma 2.

Till now we have not done anything specific topic modeling or what we call as LDA for this particular instance, what we have done is that we have used a means to represent in a format that is much more easier to handle for the model.

(Refer Slide Time: 11:27)

Notepad - Python Notebook Topic Modeling

File Edit View Insert Cell Kernel Widgets Help

Cell Toolbar Code Cell

```
[10]: [(1, 1), (1, 1), (2, 1)]
[(1, 1), (3, 1), (4, 1), (5, 1), (6, 1), (7, 1)]
[(1, 1), (2, 1), (3, 1), (4, 1), (5, 1), (6, 1), (7, 1)]
[(2, 1), (6, 1), (8, 1)]
[(3, 1), (4, 1), (7, 1)]
[(9, 1)]
[(10, 1), (10, 1)]
[(9, 1), (10, 1), (11, 1)]
[(5, 1), (10, 1), (11, 1)]
```

## LDA

```
In [10]: from gensim.corpora import Dictionary
from gensim.models import LdaModel
import numpy
%matplotlib inline
```

```
In [1]: texts = [[bank, river, shore, water],
              [river, water, flow, fast, tree],
              [bank, water, fall, wind],
              [bank, water, fall, river],
              [river, water, mud, tree],
              [money, transaction, bank, finance],
              [bank, money, company],
              [bank, finance],
              [Finance, sell, sell, bank],
              [bank, sell, sell],
              [bank, loan, sell]]
```

```
dictionary = Dictionary(texts)
corpus = [dictionary.doc2bow(text) for text in texts]
corpus
```

```
In [1]: numpy.random.seed(1) # setting random seed to get the same results each time.
model = LdaModel(corpus, id2word=dictionary, num_topics=2)
```

```
In [2]: model.show_topics()
```

```
In [3]: model.get_term_topics('water')
```

```
In [4]: model.get_term_topics('finance')
```



Now, we invoked the LDA model or the latent Dirichlet allocation model for the topic models as you remember, there are different variations of topic model that has been used as you have seen different methods in your class like the sequential mode sequential topic model and the relation topic model here we will be showing only the simple LDA model.

I import this library now let us assume another corpus where all the pre processing are initially done. So, we have something like a new corpus. So, again a tiny corpus with few sentences and the stop words are removed all the pre processing like that we convert it to the dictionary and we convert it to the bag of words format for each of this document. So, here we look at the representation for this corpus just to make things clear we have created a new dictionary based on the words in this corpus.

(Refer Slide Time: 12:35)

```
In [27]: texts = [{"bank": "river", "shoe": "water"}, {"river": "water", "flow": "fast", "tree": "fall"}, {"shoe": "water", "fall": "flow"}, {"water": "fall", "tree": "fall"}, {"river": "water", "fall": "river"}, {"river": "water", "rain": "rain"}, {"money": "transaction", "bank": "finance"}, {"bank": "money"}, {"money": "money"}, {"finance": "money", "sell": "bank"}, {"bank": "bank"}, {"bank": "loan", "sell": "sell"}]

dictionary = Dictionary(texts)
corpus = [dictionary.doc2bow(text) for text in texts]
corpus

Out[27]: [{(0, 1), (1, 1), (2, 1), (3, 1), (4, 1), (5, 1), (6, 1)}, {(0, 1), (2, 1), (3, 1), (4, 1), (5, 1), (6, 1)}, {(0, 1), (1, 1), (2, 1), (3, 1), (4, 1), (5, 1)}, {(0, 1), (2, 1), (3, 1), (6, 1)}, {(0, 1), (2, 1), (3, 1), (8, 1)}, {(0, 1), (2, 1), (3, 1), (9, 1)}, {(1, 1), (2, 1), (3, 1), (4, 1), (5, 1), (6, 1)}, {(1, 1), (2, 1), (3, 1), (33, 1)}, {(1, 1), (2, 1), (32, 1)}, {(1, 1), (2, 1), (31, 1), (12, 1), (18, 1)}, {(1, 1), (2, 1), (31, 1), (18, 1)}, {(1, 1), (2, 1), (31, 1), (13, 1)}]

In [28]: #nltk.download('punkt') # setting number need to get the same results each time.
model = LdaModel(LdaModel(corpus, id2word=dictionary, num_topics=2))

In [29]: model.show_topics()

Out[29]: [(0, u'0.164*\"bank\" + 0.142*\"water\" + 0.188*\"river\" + 0.076*\"flow\" + 0.067*\"fall\" + 0.068*\"tree\" + 0.048*\"money\" + 0.066*\"fast\" + 0.064*\"rain\"), (1, u'0.196*\"bank\" + 0.126*\"finance\" + 0.108*\"money\" + 0.082*\"sell\" + 0.067*\"river\" + 0.055*\"water\" + 0.058*\"transaction\" + 0.048*\"loan\" + 0.046*\"tree\" + 0.046*\"rain\")]
```

Now, comes to the LDA model. So, in the LDA model what we have to essentially provide to the system is the number of topics. So, we assume that the user or the programmer has a he already he is aware of the number of topics that the certain collection of corpus is going to contain and this should be provided as a parameter or as an argument to the model. Now if we see what are the other possible options that you can use to customize this particular function, you can find what are the values the hyper parameter alpha can be given; the hyper parameter eta or the beta as you know it.

The decay and number of iterations and other convenient parameters that you can use, so the alpha and eta are the important ones, others are from mostly from programming convenience. So, let us run this model. So, now, we have added this corpus into a doc 2 bow representation and we have provided that this purpose to the LDA model to get a model. This is stored in the variable named as model now if we see we can find since we have already told the system that there will be only 2 topics, it has provided 2 topics to us and it is showing the top terms that the topic represents. So, the topic is essentially a collection of words which is gathered from the different documents and you can see with what probability each word is belonging to that particular topic.

We can also find the same word might belong to multiple topics here the term bank is having a probability of 0.164 and here the term bank is having a probability of 0.196, we can find other words also repeat here like water has 0.065 in topic 1 or the topic 2 and water has a

probability of 0.142 in the first topic which is index at 0, now if you see ideally the probability should add up to 1. So, all the words which are represented here when we add up the probability it should give a probability of 1.

But if you add up these values it is not getting adding up to one. So, here we have an option to show, what are the numbers of words or topic to be shown? So, here the model has shown only top 10 words for the given topic. So, in practical scenario we often use number of topics which are more than 10 also like that it can be 50 or 100 topics in that case this function just shows 10 topics; some 10 topics out of the 100 or 50 topics now since we have a small corpus and I am aware that the number of words is going to be less than 20, I just add the parameter number of words is because the argument number of words is equal to 20 and we can find all almost all the words that is present in the particular topic if we increase the topic it does not change which means these are the only number of words in that particular topic.

(Refer Slide Time: 16:24)

File Edit View Insert Cell Kernel Widgets Help

In [31]:

```
[{0, "0.164**Bank" + 0.142**"water" + 0.100**"river" + 0.076**"flow" + 0.002**"borrow" + 0.003**"sell" + 0.000**"tree" + 0.048**"money" + 0.046**"fall" + 0.042**"rain" + 0.033**"shore" + 0.032**"land" + 0.031**"finance" + 0.021**"loan" + 0.019**"transaction" + 0.018**"loan"}, {1, "0.100**Bank" + 0.120**"finance" + 0.100**"money" + 0.082**"sell" + 0.087**"river" + 0.085**"water" + 0.065**"transaction" + 0.049**"loan" + 0.040**"tree" + 0.040**"rain" + 0.040**"shore" + 0.037**"borrow" + 0.028**"fall" + 0.027**"flow" + 0.023**"rain" + 0.021**"fall" + 0.017**"loan"}]
```

In [32]:

```
model.get_term_topics(50000)
```

In [33]:

```
[0, 0.1821230723408181, 1, 0.087247438308798511]
```

In [34]:

```
# model.get_term_topics('finance')
```

In [35]:

```
bow_water = [bank, water, bank] # bow_water = [bank, water, bank]
bow_finance = [bank, finance, bank]

bow = model.id2bow(docbow(bow_water)) # convert to bag of words format first
doc_topics, word_topics, phi_values = model.get_document_topics(bow, per_word_topics=True)
word_topics
```

Now what does that output mean? It means that like `word_type_1` our `word_type_3`, which is the word `bank`, is more likely to be in `topic_0` than `topic_1`. You must have noticed that while we unpacked into `doc_topics` and `word_topics`, there is another variable - `phi_values`. Like the name suggests, `phi_values` contains the phi values for each topic for that particular word, scaled by feature length. `Phi` is essentially the probability of that word in that document belonging to a particular topic. The next few lines should illustrate this.

In [36]:

```
phi_values
```

This means that `word_type_0` has the following `phi_values` for each of the topics. What is interesting to note is `word_type_0` - because it has 2 occurrences (i.e. the word `bank` appears twice in the bow), we can see that the scaling by feature length is very evident. The sum of the `phi_values` is 2, and not 4.

In [37]:

```
bow = model.id2bow(docbow(bow_finance)) # convert to bag of words format first
doc_topics, word_topics, phi_values = model.get_document_topics(bow, per_word_topics=True)
word_topics
```

In [38]:

```
all_topics = model.get_document_topics(corpus, per_word_topics=True)
```

Now, if we want to know what is the probability of a particular word? So, we get the term topics from this function. So, we can find that water has a probability of 0.128 and a 0.047 in either of the topics, now let us see a new document that is bank; water bank which was not provided to the model while it was learning the probability distributions. So, we have a new test document or a new document where we have bank water bank and we first convert it to the b o w representation, once we have that here what we find is that the word 0 and word 3. So, both the terms which are like bank and water both the terms are more likely to belong to

topic 0. So, here even if bank has a higher probability at topic one may be because of it is co occurrence with water which has a high score in topic 0 both bank and water are given a higher probability of being in topic 0 for this particular document.

As you are aware in topic models given a word it can belong to multiple topic distributions and once you find the particular word in the document it might be coming from any one of those topics.

(Refer Slide Time: 18:22)

```

In [18]: numpy.random.seed(1) # setting random seed to get the same results each time
model = LdaModel(id2word=id2word, id2word=id2word, num_topics=2)

In [19]: model.show_topics(num_words=10)
Out[19]: [(0, '0.1487*"bank" + 0.1427*"water" + 0.1887*"river" + 0.0767*"flood" + 0.0637*"corro" + 0.0637*"salt" + 0.0607*"tree" + 0.0487*"money" + 0.0487*"fast" + 0.0447*"rain" + 0.0427*"fall" + 0.0337*"shore" + 0.0317*"oil" + 0.0317*"finance" + 0.0257*"loan" + 0.0197*"transaction" + 0.0197*"land"),
           (1, '0.3567*"bank" + 0.1267*"finance" + 0.0837*"water" + 0.0627*"river" + 0.0657*"water" + 0.0567*"transaction" + 0.0487*"loan" + 0.0467*"tree" + 0.0467*"oil" + 0.0467*"shore" + 0.0377*"corro" + 0.0287*"fall" + 0.0277*"flood" + 0.0257*"rain" + 0.0237*"fast" + 0.0177*"land")]

In [20]: model.get_term_topics('water')
Out[20]: [(0, 0.12821234073249418), (1, 0.047247418568794511)]

In [21]: model.get_term_topics('Finance')
Out[21]: [(0, 0.12821234073249418), (1, 0.047247418568794511)]

In [22]: bow_water = LdaModel.doc2bow(bow_water) # convert to bag of words format first
doc_topics, word_topics, phi_values = model.get_document_topics(bow, per_word_topics=True)

word_topics
Out[22]: [(0, [0, 1]), (1, [0, 1])]

Now what does that output mean? It means that the word_type_1, our word_type_3, which is the word bank, is more likely to be in topic_0 than topic_1.
You must have noticed that while we unpacked into doc_topics and word_topics, there is another variable - phi_values. Like the name suggests, phi_values contains the phi values for each topic for that particular word, scaled by feature length. Phi is essentially the probability of that word in that document belonging to a particular topic. The next few lines should illustrate this.

In [23]: phi_water
Out[23]: [(0, [0, 0.248465364294323]), (1, 0.071313444337986094)]
          [(1, 0.981712087307244), (0, 0.418287902027564)]]

This means that word_type_0 has the following phi_values for each of the topics. What is interesting to note is word_type_3 - because it has 2 occurrences (i.e. the word bank appears twice in the bow), we can see that the scaling by feature length is very evident. The sum of the phi_values is 2, and not 1.

```

Now, if you see here the 5 values which is essentially the probability of a word in that document belonging to a particular doc topic. So, the 5 value essentially shows, what is the probability of each word belonging to each of those topics? So, if you see here the word type 3 when you add up the values here it has a probability it shows a score not a probability, but a score which adds up to 2.

If the sum of the 5 value is 2 which is and not 1 and this is indicative of the scaling by feature length. So, this always does not happen, but it is an indicative aspect here now what we can see here is that we have taken the second document here which is b o w finance. So, once we have bank finance bank as our document we can find that both the words bank and finance have a higher probability of belonging to topic 1. So, here if you see the word 3 which was bank had a higher probability of belonging to topic 0.

(Refer Slide Time: 19:41)

```
In [34]: phi_values
Out[34]: [(0, [0, 0.9248040556429432]), (1, 0.075155444357056894)]
[(0, [0, 1.981712097307244]), (1, 0.41826790295275661)]

This means that word_type 0 has the following phi_values for each of the topics. What is interesting to note is word_type 0 - because it has 2 occurrences (i.e. the word sunk appears twice in the box), we can see that the scaling by feature length is very evident. The sum of the phi_values is 2, and not 1.

In [35]: bow = model_lda.bow(doc2bow[bow_finance]) # convert to bag of words format first
doc_topics, word_topics, phi_values = model.get_document_topics(bow, per_word_topics=True)
word_topics
Out[35]: [(0, [1, 0]), (1, [1, 0])]

In [36]: all_topics = model.get_document_topics(corpus, per_word_topics=True)
for doc_topics, word_topics, phi_values in all_topics:
    print('New Document')
    print('Document topics:', doc_topics)
    print('Word topics:', word_topics)
    print('Phi values:', phi_values)
    print('-----')

New Document
Document topics: [(0, [0, 1]), (1, [0, 1]), (2, [0, 1]), (3, [0, 1])]
Word topics: [(0, [0, 1]), (1, [0, 1]), (2, [0, 1]), (3, [0, 1])]
Phi values: [(0, [(0, 0.00370873879809255), (1, 0.094210126320801919)]),
(1, [(0, 0.00370873879809255), (1, 0.094210126320801919)]),
(2, [(0, 0.00370873879809255), (1, 0.094210126320801919)]),
(3, [(0, 0.00370873879809255), (1, 0.094210126320801919)])
-----
```

But in the second document, this is the model assumes that the word bank comes from topic 1 not topic 0, this is again assumed to be because it co occurs with the word finance. So, we should be talking about the financial institution bank rather than the river side or river bank. So, this becomes evident using this model.

(Refer Slide Time: 20:08)

```
In [36]: bow = model_lda.bow(doc2bow[bow_finance]) # convert to bag of words format first
doc_topics, word_topics, phi_values = model.get_document_topics(bow, per_word_topics=True)
word_topics
Out[36]: [(1, [1, 0]), (2, [1, 0])]

In [37]: all_topics = model.get_document_topics(corpus, per_word_topics=True)
for doc_topics, word_topics, phi_values in all_topics:
    print('New Document')
    print('Document topics:', doc_topics)
    print('Word topics:', word_topics)
    print('Phi values:', phi_values)
    print('-----')

New Document
Document topics: [(0, 0.00370873879809255), (1, 0.094210126320801919)]
Word topics: [(0, [0, 1]), (1, [0, 1]), (2, [0, 1]), (3, [0, 1])]
Phi values: [(0, [(0, 0.00370873879809255), (1, 0.094210126320801919)]),
(1, [(0, 0.00370873879809255), (1, 0.094210126320801919)]),
(2, [(0, 0.00370873879809255), (1, 0.094210126320801919)]),
(3, [(0, 0.00370873879809255), (1, 0.094210126320801919)])
-----
```

Now, let us see how what are the different values that we get for each of these documents. So, we can find that the document topics. So, the first document which is essentially bank river shore water if the document representation has a higher tendency to or has a higher

proportion of topic 0 rather than topic 1.

Now, individual words were all the 4 words 0, 1, 2 and 3 have a higher tendency to belong the topic 0 the 5 values which is like per the per word distribution for each word in that document also shows the similar branch. Now if you see next word per document which is the document river water flow fast tree what we can find is that again it is having a higher probability of belonging to topic 0 than topic one and we can find that some of the words have no, no occurrence or no probability of occurrence in topic one like the 4 and 6.

(Refer Slide Time: 21:28)

```
Document topics: [(0, 0.87888826668878147), (1, 0.1213107333121853)]
word topics: [(0, [0, 1]), (2, [0, 1]), (4, [0, 1]), (6, [0, 1]), (8, [0, 1])]
Phi values: [(0, [(0, 0.9781001240171413), (1, 0.02108091708761819)]), (2, [(0, 0.91271370180360953), (1, 0.06720020810433072
4)]), (4, [(0, 0.8881003880274291), (1, 0.0180830071607817)), (6, [(0, 0.97541485007871162), (1, 0.034585940925120051)])]

*****
```

```
New Document:
Document topics: [(0, 0.87888826668878147), (1, 0.1213107333121853)]
word topics: [(0, [0, 1]), (2, [0, 1]), (4, [0, 1]), (6, [0, 1]), (8, [0, 1])]
Phi values: [(0, [(0, 0.9781001240171413), (1, 0.02108091708761819)]), (2, [(0, 0.96508226710042713), (1, 0.0341773200357287
9)]), (4, [(0, 0.8514204600007707), (1, 0.1400791391902921)]), (6, [(0, 0.97846429434929493), (1, 0.021535795758705195)])]

*****
```

```
New Document:
Document topics: [(0, 0.8504902055107282), (1, 0.143519734480027172)]
word topics: [(0, [0, 1]), (2, [0, 1]), (4, [0, 1]), (6, [0, 1]), (8, [0, 1])]
Phi values: [(0, [(0, 0.9707500870318059), (1, 0.02923009124045501)]), (2, [(0, 0.958388012422396477), (1, 0.04101047577883408
9)]), (4, [(0, 0.91008681343132348), (1, 0.040151432794878299)]), (6, [(0, 0.90008014000421006), (1, 0.0001040093315810993)])]

*****
```

```
New Document:
Document topics: [(0, 0.115495278737779), (1, 0.8889044721202229)]
word topics: [(0, [0, 1]), (2, [0, 1]), (4, [0, 1]), (6, [0, 1]), (8, [0, 1])]
Phi values: [(0, [(0, 0.840050873161608174), (1, 0.05994112561009174)]), (10, [(0, 0.02010213651685854), (1, 0.9700790504004971)]), (12, [(0, 0.9100868135394871), (1, 0.04174461008716177))]

*****
```

```
New Document:
Document topics: [(0, 0.4433000011307140763), (1, 0.5560014800240239)]
word topics: [(0, [1, 0]), (10, [1, 0]), (11, [0, 1]), (12, [0, 1])]
Phi values: [(0, [(0, 0.889200139297447), (1, 0.010079180078020529)]), (10, [(0, 0.23450000405520753), (1, 0.7654030050447952
9)]), (12, [(0, 0.400015520000044), (1, 0.043204617000044)])]
```

Now, if you look into say then one of those documents where the it is probability of belonging to topic 1 is 0.88 while that of belonging to topic 0 is 0.11 and we can find almost all of the words have a higher probability to belong to a topic one it is also possible that suppose similarly for this document.

(Refer Slide Time: 22:01)

```
Document topics: [(0, 0.23860000000000002), (1, 0.12110000000000001)]
word topics: [(0, [0, 1]), (1, [0, 1]), (2, [0, 1]), (3, [0, 1]), (4, [0, 1])]
Phi values: [(0, [(0, 0.9700415861273085), (1, 0.02008413807263115)]), (1, [(0, 0.9500822071864271), (1, 0.03431772880732879)]), (2, [(0, 0.10342098468009700), (1, 0.1480701530009232)]), (3, [(0, 0.7940432493249300), (1, 0.21535705750705195)])]

New Document

Document topics: [(0, 0.45648000000000005), (1, 0.14510176480000002)]
word topics: [(0, [0, 1]), (1, [0, 1]), (2, [0, 1]), (3, [0, 1]), (4, [0, 1]), (5, [0, 1])]
Phi values: [(0, [(0, 0.9700763008785580), (1, 0.02930091314841355)]), (1, [(0, 0.9500851422106477), (1, 0.04150477700340085)]), (2, [(0, 0.05000451241921208), (1, 0.04033545758087296)]), (3, [(0, 0.000081405041006), (1, 0.0900108533583093)])]

New Document

Document topics: [(0, 0.11540552787327779), (1, 0.8845044723162223)]
word topics: [(0, [1, 0]), (1, [0, 1]), (2, [1, 0]), (3, [0, 1]), (4, [1, 0]), (5, [0, 1])]
Phi values: [(0, [(0, 0.990015253914771), (1, 0.0100001525391477)]), (1, [(0, 0.820102136501605854), (1, 0.0790978034083333)])
(2, [(0, 0.0930015253914771)]), (3, [(0, 0.01744810087611777)]))

New Document

Document topics: [(0, 0.44300051307140975), (1, 0.5500011480028002295)]
word topics: [(0, [1, 0]), (1, [0, 1]), (2, [0, 1]), (3, [0, 1]), (4, [0, 1])]
Phi values: [(0, [(0, 0.78300200181290787), (1, 0.02007918007802029)]), (1, [(0, 0.234506008405240763), (1, 0.7654000050847923)])
(2, [(0, 0.0500013200000004), (1, 0.341000007000004)]))

New Document

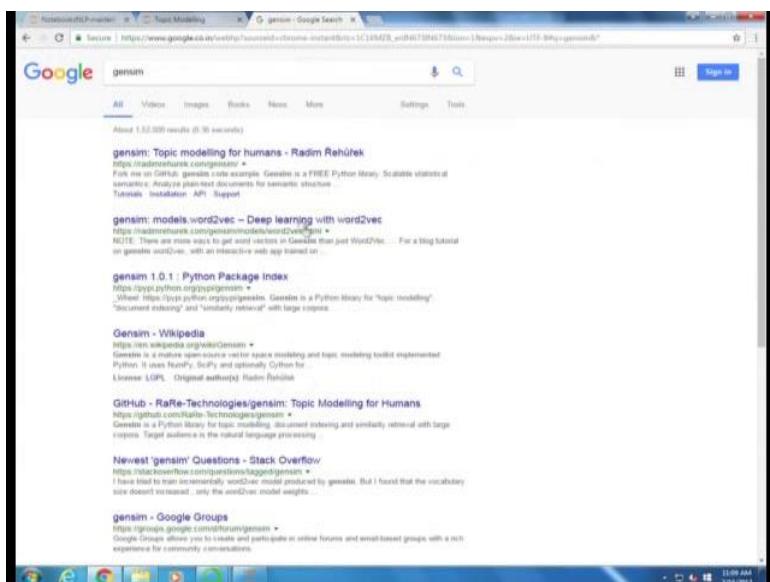
Document topics: [(0, 0.01813547546016004), (1, 0.7080002245307306)]
word topics: [(0, [1, 0]), (1, [0, 1]), (2, [1, 0]), (3, [0, 1]), (4, [1, 0])]
Phi values: [(0, [(0, 0.000040191810077283), (1, 0.113058000000012193)]), (1, [(0, 0.018030405180000102), (1, 0.00110000401000196)])]

New Document

Document topics: [(0, 0.12500028184008439), (1, 0.874000018150001587)]
word topics: [(0, [1, 0]), (1, [0, 1]), (2, [1, 0]), (3, [0, 1]), (4, [1, 0]), (5, [0, 1]))]
```

We have more or less similar weights for each of the topic like topic 0 it is 0.44 and topic 1, it is 0.55 where the confidence of this document belong to a particular topic is not that high to be decisive enough we can find that 2 of the words have a probability of belonging to topic 1 while that last third word has a higher probability of belonging to topic 0 and we can find the comparative values of each of those doc words with this topic proportion. So, this is how we analyze a corpus using topic modeling or the LDA.

(Refer Slide Time: 22:55)



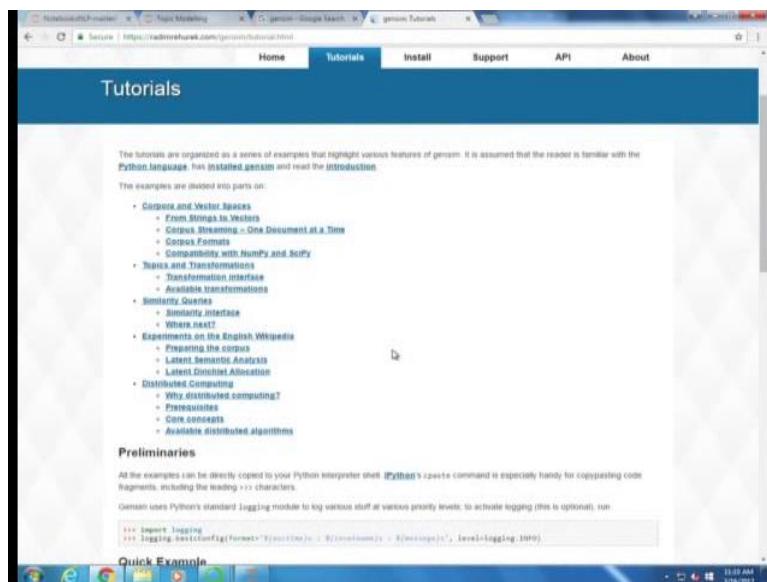
Gensim essentially provides a suit of packages that helps in different topic modeling task and

they have a set of comprehensive tutorials. So, the tutorial that you have been seen here has been combined from multiple tutorials from Gensim.

(Refer Slide Time: 23:11)



(Refer Slide Time: 23:13)



You can have a further look into different topics here, then in addition to LDA provide other topic models as well they have they show topic models e pitching load HDP which is hierarchical display process and they also have a model for sequential LDA. So, we can have a quick look into one of those.

(Refer Slide Time: 23:51)

```

<>> for doc in corpus[1]: # here <tfdif> and <tfid> transformations are actually executed here, on the fly
[(0, -0.095), (1, 0.529) # "human machine interface for lab abc computer applications"
(0, 0.095), (2, 0.723) # "user interface design at the human-computer-response time"
(0, -0.095), (2, 0.632) # "system and human system engineering testing of bms"
(0, 0.095), (3, 0.529) # "computer systems reliability assessment and measurement"
(0, -0.095), (3, -0.165) # "The generation of random binary unordered trees"
(0, 0.095), (3, -0.145) # "graph minors IV widths of trees and well quasi ordering"
(0, -0.095), (3, 0.084) # "Graph minors a survey"
]

Model persistency is achieved with the <save()> and <load()> functions:

<>> lsi.save('/tmp/model.lis') # save for <tfdif>, <ids> ...
<>> lsi = models.LsiModel.load('/tmp/model.lis')

The next question might be just how exactly similar are those documents to each other? Is there a way to formalize the similarity, so that for a given input document, we can order some other set of documents according to their similarity? Similarity queries are covered in the next tutorial

Available transformations

Gensim implements several popular vector space model algorithms:

```

- Term Frequency / Inverse Document Frequency, TfIdf**: expects a bag-of-words (integer values) training corpus during initialization. During transformation, it will take a vector and return another vector of the same dimensionality, except that features which were rare in the training corpus will have their value increased. It therefore converts integer-valued vectors into real-valued ones, while leaving the number of dimensions intact. It can also optionally normalize the resulting vectors to (Euclidean) unit length;

```

<>> model = models.TfidfModel(corpus, normalize=True)

```

- Latent Semantic Indexing, LSI (or sometimes LSA)**: transforms documents from either bag-of-words or (preferably) Bag-of-words into a latent space of a lower dimensionality. For the toy corpus above we used only 2 latent dimensions, but on real corpora, target dimensionality of 200-500 is recommended as a "golden standard" [1]

```

<>> model = models.LsiModel(fldf_corpus, Ldbow=500, num_topics=300)

LSI training is unique in that we can continue "training" at any point, simply by providing more training documents. This is done by incremental updates to the underlying model in a process called online training, because of this feature, the input document stream may even be infinite - just keep feeding LSI new documents as they arrive, while using the computed transformation mode as read-only in the meantime!

```

```

<>> model.add_documents(other_tfldf_corpus) # now lsi has been trained on tfldf_corpus + another_tfldf_corpus
<>> lsi_vec = model[tfldf_vec] # convert some new document into the LSI space, without affecting the model
<>> model.add_documents(more_documents) # tfldf_corpus + another_tfldf_corpus + more_documents
<>> lsi_vec = model[tfldf_vec]

```

(Refer Slide Time: 23:48)

```

• Random Projections, RP aim to reduce vector space dimensionality. This is a very efficient (both memory- and CPU-bound) approach to approximating TfIdf distances between documents. By "flattening" a 10M+ document dataset, Recommended target dimensionality is again in the hundreds/thousands, depending on your dataset.

<>> model = models.RpModel(tfldf_corpus, num_topics=100)

• Latent Dirichlet Allocation, LDA is yet another transformation from bag-of-words counts into a topic space of lower dimensionality. LDA is a probabilistic extension of LSA (also called multinomial PCA), so LDA topics can be interpreted as probabilistic distributions over words. These distributions are, just like with LSA, inferred automatically from a training corpus. Documents are in turn interpreted as a (soft) mixture of these topics (again, just like with LSA)

<>> model = models.LdaModel(corpus, Ldbow=500, num_topics=100)

gensim uses a fast implementation of online LDA parameter estimation based on [2], modified to run in distributed mode on a cluster of computers

• Hierarchical Dirichlet Process, HDP is a non-parametric Bayesian method (note the missing number of required topics)

<>> model = models.HdpModel(corpus, Ldbow=500, num_topics=100)

gensim uses a fast, online implementation based on [3]. The HDP model is a new addition to gensim, and still rough around its academic edges - user with care

Adding new VSM transformations (such as different weighting schemes) is rather trivial, see the API reference or directly the Python code for more info and examples

It is worth repeating that these are all unique incremental implementations, which do not require the whole training corpus to be present in main memory at once. VSM memory taken care of, too: I am now improving Distributed Computing to improve CPU efficiency, too. If you feel you could contribute (by testing, providing use-cases or code), please let me know

Continue on to the next tutorial on Similarity Queries

[1]Brodley, 2008. An empirical study of required dimensionality for large-scale latent semantic indexing applications.
[2]Hofmann, Blei, Bach, 2010. Online learning for Latent Dirichlet Allocation.
[3]Wang, Paisley, Blei, 2011. Online variational inference for the Hierarchical Dirichlet process.
[4]Halko, Martinsson, Tropp, 2009. Finding structure with randomness.
[5]Rensink, 2011. Topicpace tracking for Latent Semantic Analysis.

```



Home | Tutorials | Install | Support | API | About | **Support:**

Stay informed via gensim mailing list

Here we can find different transformations or different ways of representing a document what DF, IDF and LSA or LSI are some of the pops they use to be some of the popular ways of representing a document in a vectorial representation and we have LDA which is what we have seen we have HDP. So, HDP is a non parametric version where you do not need to provide the number of topics as well that is it is inferred by the model itself. So, this is pretty much for the topic modeling tutorial.

Thank you.

**Natural Language Processing**  
Prof. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 59**  
**Tutorial**

Hello everyone, welcome to the tutorial section for the NLP course. So, today we will be looking into how to use or how to build a classifier for one of the most popular task in NLP sentiment analysis and we will be looking to how we create features and how we can build a supervised machine learning classifier for the purpose of sentiment analysis.

(Refer Slide Time: 00:24)



The screenshot shows a Jupyter Notebook interface with the title "Sentiment Analysis - NaiveBaye's, SVM, Random Forests". The notebook contains a single cell with the following Python code:

```
In [1]: import os
import collections
import Counter
f = open('positive.txt')
positive_reviews = []
for line in f:
    positive_reviews.append(line)
f.close()
positive_text = "\n".join(positive_reviews)

f = open('negative.txt')
negative_reviews = []
for line in f:
    negative_reviews.append(line)
f.close()
negative_text = "\n".join(negative_reviews)

# calculate word counts
positive_word_counts = Counter.Counter(positive_text.lower().split())
negative_word_counts = Counter.Counter(negative_text.lower().split())

# calculate probabilities
positive_probabilities = {}
negative_probabilities = {}

for word in positive_word_counts:
    positive_probabilities[word] = (positive_word_counts[word] + 1) / (len(positive_reviews) + len(positive_word_counts))

for word in negative_word_counts:
    negative_probabilities[word] = (negative_word_counts[word] + 1) / (len(negative_reviews) + len(negative_word_counts))
```

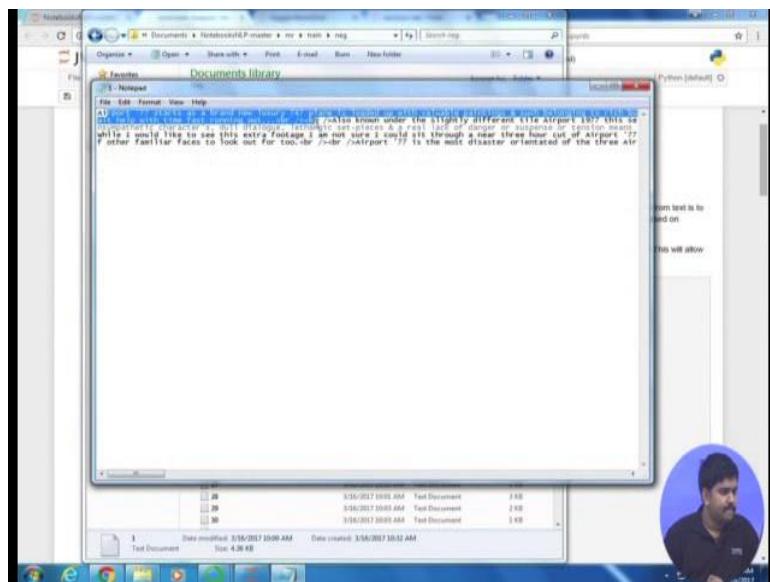
So, we will essentially look into how we can build a Naive Bayes classifier and after that we will just give you pointers towards how to build other classifiers. The notebook provided to you will have code snippets for building other type of classifiers, I will not be discussing it here.

So, sentiment analysis is essentially a task where you are given a document which can be a sentence or a set of sentences and you are given with and you have to label it whether it belongs to a positive sentiment or a negative sentiment. This is the most simplistic representation or a task that we can think of sentiment analysis. Of course, as per the demands, this complexity of the task can be increased or like depending on the requirement

from the customer or the user, there might be other levels of sentiment that might be applied here. So, here what we will be doing is that we will be taking about 5000 documents which are 5000 movie reviews and we see and we are already provided with the labels for those 5000 documents.

Here we can find about 2500 negative reviews and positive reviews about different films.

(Refer Slide Time: 02:05)



Now once this are provided, what we have to do is that we have to build a supervise classifier where we tag the positives as 1 and the negatives as 0 and then we build the classifier. So, here each document is your input and your models have to predict the sentiment of a new document when provided in the similar representation. So, we have to first convert each of those documents which is a training document in your vectorial representation. So, for that purpose, what we do here is that we take each positive and negative documents into separate data structure which is basically a list.

(Refer Slide Time: 03:28)



```
#import file
f = open("nr-train/neg/0.txt", "r")
for line in f:
    print line
f.close()

positive_text = " ".join(postrviews)
negative_text = " ".join(negrevines)

#Count word counts for negative time.
neg_counts = Counter(negative_text)
#Generate word counts for positive time.
positive_counts = Counter(positive_text)

print("Negative text sample: (%d)" %format(negative_text[:500]))
print("Positive text sample: (%d)" %format(positive_text[:500]))
```

Negative text sample: Story of a man who has unnatural feelings for a pig. Starts out with a opening scene that is a terrific example of absurd comedy. A formal orchestra audience is turned into an insane, violent mob by the crazy chartings of it's singer. Unfortunately it stays absurd the WHOLE time with no general narrative eventually making it just too off putting. Even though the arc should be turned off. The cryptic dialogue would make Shakespeare seem easy to a third grader. On a technical level it's a masterpiece.

Positive text sample: Brummell High is a cartoon comedy. It ran at the same time as some other programs about school life, such as "Teachers". My 16 years in the teaching profession lead me to believe that Brummell High's satire is much closer to reality than "Teachers". The scrabbles to survive financially, the insightful students who can see right through their pathetic teachers' pangs, the pettiness of the whole situation, all remind me of the schools I knew and their students. When I saw the episode in which a s

And once we do that what we do is that we count the number of individual words that is appearing in each of these sets.

We can find that since each document here is given as a separate file, we use the inbuilt OS library to traverse through each of the file given in the folder pos or the positive and then we append the content to a particular list. Once we do that we define a function called count text which basically looks for individual words in each of the document and also it keeps a track of the count or the number of time it is appearing.

(Refer Slide Time: 04:39)



```
#Count word counts for negative time.
neg_counts = Counter(negative_text)
#Generate word counts for positive time.
positive_counts = Counter(positive_text)

print("Negative text sample: (%d)" %format(negative_text[:500]))
print("Positive text sample: (%d)" %format(positive_text[:500]))
```

In [7]: negative\_counts

```
{'theseebe': 1,
 'theebe': 2,
 'unidy': 1,
 'comically': 2,
 'comicaly': 1,
 'harmless-looking': 2,
 'LAST': 9,
 'hell': 1,
 '...': 1,
 'print...': 1,
 'unihed': 2,
 '...': 5,
 '/other': 9,
 'word!': 1,
 'word!': 1,
 'word.' 2,
 'screeching': 22,
 'pig': 4,
 'jacksones': 1,
```

Making Predictions

Note that we have the word counts, we just have to convert them to probabilities and multiply them out to get the predicted classification. Let's say

We can just have a look of how the variable negative\_counts looks like, if you see here for negative the words pineapple, consider, daring, etcetera are appearing in different counts. So, we can find western appearing 20 times, screening appear 22 times, wooden appears 42 times and so on. So, here we have not done any pre processing for as it is so, far illustrative purposes. So, in this corpus may be the daring might be at separate word than this word because it has some extra symbols in it. These all need to be taken care when we go for a practical application. We do not do any case folding or any removal of other special symbols, but all these needs to be taken care.

So, once we have this separate dictionaries, now what we do is do here is that we just find out, how often a word is appearing in both positive and negative context. It is often possible that some of these words can appear in positive and negative context. So, the very basic idea here is that if a word appears more or less like equally in both the sets, they does not have much value, if a word words presents in one of these sets say positive context is queued with compared to what is happening with the negative set then that is more likely a positive sentiment word. If a word is more likely to belong to a negative set it is more likely a negative sentiment word, if it is more likely to belong to a positive set, it is a positive word.

(Refer Slide Time: 06:26)

```

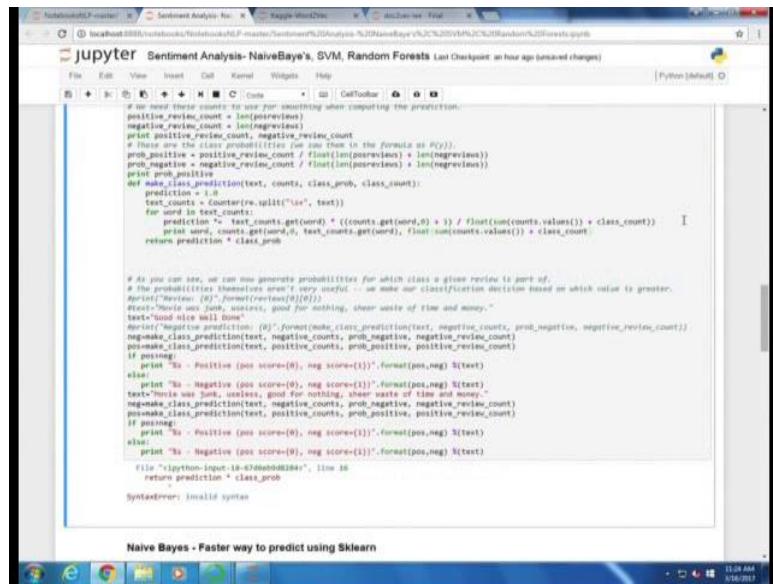
In [4]: # we need these counts to use for smoothing when computing the prediction.
positive_review_count = len(positive_reviews)
negative_review_count = len(negative_reviews)
# these are the class probabilities (as you know in the formula as P(y)).
prob_positive = positive_review_count / float(len(positive_reviews)) + len(negative_reviews)
prob_negative = negative_review_count / float(len(positive_reviews)) + len(negative_reviews)

def make_class_predictions(text, counts, class_prob, class_count):
    # first we split the text into words
    text_counts = Counter(re.split("\s+", text))
    for word in text_counts:
        # for each word in the text, we get the number of times that word occurred in the review for a given class, add 1 to smooth
        # Counting ensures that we don't multiply the prediction by 0 if the word didn't exist in the training data.
        # an alias smooth the denominator counts to keep things even
        # a prior "1d, 1d, 1d" (text_counts.get(word, 0), counts.get(word, 0), sum(counts.values())))

```

We have the description here where we what we do is that we would find the total number of times a particular word that occur in the corpus and we just find the probability of it with its total count.

(Refer Slide Time: 06:50)



The screenshot shows a Jupyter Notebook interface with a Python script titled "Naive Bayes - Faster way to predict using Sklearn". The code implements a Naive Bayes classifier for sentiment analysis. It starts by importing necessary modules and defining helper functions for calculating class probabilities and making predictions. The main loop reads text from standard input, tokenizes it, and calculates individual word probabilities. These are then combined with the overall positive and negative review counts to produce a final prediction. The output shows the predicted class ("positive" or "negative") and its probability score.

```
# we need these counts to use for smoothing when computing the prediction.
positive_review_count = len(positive_reviews)
negative_review_count = len(negative_reviews)
prob_positive = positive_review_count / float(len(positive_reviews) + len(negative_reviews))
prob_negative = negative_review_count / float(len(positive_reviews) + len(negative_reviews))
print prob_positive

def make_class_prediction(text, counts, class_prob, class_count):
    prob = class_prob
    text_counts = Counter(text.split(" "))
    for word in text_counts:
        print word, counts.get(word, 0) * ((counts.get(word, 0) + 1) / float(counts.values()[-1] + class_count))
    return prediction + class_prob

# As you can see, we can now generate probabilities for which class a given review is part of.
# The probabilities themselves aren't very useful, we make our classification decision based on which value is greater.
print("Review: (%s)"%text)
print("Movie was just okies, good for nothing, sheer waste of time and money."%text)
text="Movie was just okies, good for nothing, sheer waste of time and money."
make_class_prediction(text, negative_counts, prob_negative, negative_review_count)
make_class_prediction(text, positive_counts, prob_positive, positive_review_count)

if posneg:
    print "%s - Positive (pos score=%f, neg score=%f)"%format(pos,neg) %text
else:
    print "%s - Negative (pos score=%f, neg score=%f)"%format(pos,neg) %text
File "<ipython-input-18-670d0e00d8284c". line 18
      return prediction + class_prob
SyntaxError: invalid syntax
```

If we see what exactly is prob positive. So, what we do here is that we took to represent the notion of cuteness or more likelihood of something being occurring in one set than the other, we convert them to a suitable probability distribution and what we find here is that the positive review count and negative review count. They are more or less equal they are like roughly 2500 a little bit here and there, but roughly they are equal. So, yes we can find what is the respective count.

We can find there is a slight difference of 300 also between both of them, but we essentially have more or less 2500 documents in each of those classes. Now so we can roughly say that they are like near to 0.5 in probability each, there is the number of documents that has prob positive sentiment and negative sentiment and are roughly half once we have that what remains is to find the individual probability of each word that is present. So, in order to do that, let us see, how this work, how this is calculated? So, print word and we will also find count, it seems to have a minor error in our work. So, indentation is something that is pretty important in python.

(Refer Slide Time: 11:41)

The screenshot shows a Jupyter Notebook interface with two panes. The top pane displays the output of a code cell, which prints a list of words and their associated probabilities for both positive and negative classes. The bottom pane shows the code used to perform the Naive Bayes classification.

```
well 1 8.3490547153e-05
well 1 8.3490547153e-05
Done 1 8.3493641243e-08
Done 1 8.3493641243e-08
well 1 8.08313807803e-05
well 1 8.08313807803e-05
nice 1 0.00029890723523
Good nice well Done - Positive (pos score=8.0643230824e-18, neg score=1.5366081111e-19)
and 1 0.00018151240001e-05
sheer 1 4.58457314072e-05
good 1 0.00010081270808
money 1 0.00010081270808
waste 1 0.00047298078776
for 1 0.0070556140642
nothing 1 0.207130432736e-06
time 1 9.00012104327362
useless 1 4.58457314072e-06
of 1 0.01151000000000001e-05
nothing 1 0.00018151240001e-05
Junk 1 0.00008618753e-06
was 1 0.0001840577240
and 1 0.0001840577240
sheer 1 4.67564390308e-05
good 1 0.001120043175
money 1 0.001120043175
waste 1 2.675700515108e-05
for 1 0.00090903127045
nothing 1 0.00018151240001e-05
time 1 0.00112422839317
useless 1 1.60007283249e-06
of 1 0.00018151240001e-05
nothing 1 2.33782195148e-06
Junk 1 1.669973232349e-06
use 1 0.0001712162798
Movie was junk useless good for nothing, sheer waste of time and money - http://tiny.cc/meyarw (pos score=3.20700245757e-47, neg score=1.38952285531e-44)
```

```
In [1]: from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split
from sklearn import metrics
test=[]
actual=[]
for filename in os.listdir("mr/test/pos"):
    #open('mr/test/pos/%s' % filename, "r")
    f = open('mr/test/pos/%s' % filename, "r")
    test.append(f.read())
    f.close()
for filename in os.listdir("mr/test/neg"):
    #open('mr/test/neg/%s' % filename, "r")
    f = open('mr/test/neg/%s' % filename, "r")
    test.append(f.read())
    f.close()
```

Here what we can find is that we can find the probability of each word belonging to a particular class. So, the function is defined in such a way that we first calculate all the prob, all the probability, all the words probability of belonging to the negative class then we find the probability of each of the word in the same document to belong to the positive class. So, suppose our new sentence is good, nice, well and done, we will calculate first probability of those words belonging to the negative counts and for the negative probability set and then to the positive probability set.

If we see here, so done, good, well and nice, first when they are given to the negatives, we are getting a probability of 3.003 into 10 to the power of minus 6 as compared to 8.34 into 10 to the power minus 6, similarly for other values also like good negative has a probability of appearing with 4.5 into 10 to the power minus 5 where is to good in positive is higher than that. So, once we have this individual scores it take the product of this course as shown here and we find that the positive score for this document which convince this 4 words is 8.06 into the power 10 power minus 8 as compared to the negative score which is much lesser than this. So, we can say that overall sentiment for this review is positive.

Now let us feed for another sentence where the sentence is the movie was junk useless good for nothing sheer waste of time and money. So, it does not take much to think that what will be the label for this document, but what we have to see is that whether the system is able to capture that see if you find the sentiment of this particular document is negative and we can

find that this particular document as a very huge difference in its positive score and negative score. We can find that the value here is in the order of 10 to the power minus 44 while the positive score is somewhere at in the value of 10 to the power of minus 47.

So, this means this particular document is labeled as negative and the system is able to capture both the cases of positive and negative documents are released for these 2 test document cases. So, this is how we internally calculate the values.

(Refer Slide Time: 15:01)

```
In [3]: from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import CountVectorizer
from sklearn import metrics
text = []
actual = []
for filename in os.listdir("nr/test/pos"):
    f = open("nr/test/pos/" + filename, "r")
    str = f.read()
    text.append(str)
    f.close()
    actual.append(1)
for filename in os.listdir("nr/test/neg"):
    f = open("nr/test/neg/" + filename, "r")
    str = f.read()
    text.append(str)
    f.close()
    actual.append(-1)
reviews = []
for r in reviews:
    reviews.append(r)
    reviews.append(r)
    reviews.append(r)
# Compute counts from text using a vectorizer. There are other vectorizers available, and lots of options you can set.
# This one removes stop words and uses binary features.
vectorizer = CountVectorizer(stop_words='english')
X_train, X_test, y_train, y_test = train_test_split(reviews, actual, test_size=0.2, random_state=42)
# Train the model
model = MultinomialNB().fit(X_train, y_train)
# Predict
predicted = model.predict(X_test)
# Compute metrics
print(metrics.classification_report(y_test, predicted))
print(metrics.confusion_matrix(y_test, predicted))
print(metrics.accuracy_score(y_test, predicted))
print(metrics.precision_score(y_test, predicted))
print(metrics.recall_score(y_test, predicted))
print(metrics.f1_score(y_test, predicted))

# Now we can use the model to predict classifications for our test features.
predictions = model.predict(text_features)
print(predictions)

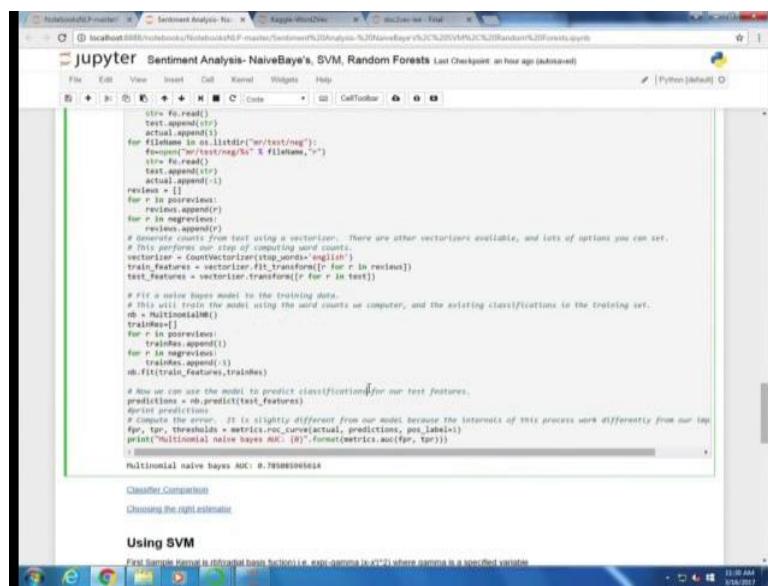
# Print predictions
for i in range(len(predictions)):
    print("Actual: " + str(actual[i]) + " Predicted: " + str(predictions[i]))
    if actual[i] == predictions[i]:
        tpr, fpr, thresholds = metrics.roc_curve(actual[i], predictions[i], pos_label=1)
        print("Multinomial naive bayes AUC: (%.3f)" % metrics.auc(tpr, fpr))
```

Now, how we can use a Naive Bayes; Naive Bayes classifier for a production ready purpose. So, here what we do is that we use the Scikit learn library they have a Naive Bayes classifier already implemented what we will do is that we will use this classifier to do the same task that is shown here, but we test it on a larger number of test data. So, while giving he input to the system we have to again convert it into a suitable vectorial representation that this particular library understands. So, we use the count vector is a function which basically represents a document in terms of the number of times a particular word is appearing in that document and it by default removes the stop words or it has other possible functions as well.

This particular function provides a whole range of facilities that makes our job much easier like we can often use different multi word patterns for multi word ranges like for example, if you have a word like hot dog though they are essentially separated by space, hot dog does not, when we take the individual words hot or dog, it does not capture the notion that is represented by hot dog. So, it is assumed to be a multi word expression. So, we will be; so

these kind of Engram range functionality will help you to capture those multi word expressions and of course, we have to use some filtering criteria to remove the unnecessary ones and we convert both the training vectors and the test vectors into this vectorial representation. So, there is again a small difference in the function called that we are doing fit transform and transform. So, fit and transform are actually two different methods and this Scikit learn function fit transform convert basically performs both the functions together. So, in fit we have to first build the vocabulary or the unique word.

(Refer Slide Time: 16:48)



```

# jupyter Sentiment Analysis- NaiveBayes, SVM, Random Forests Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help CellToolbar Python (default) O

# Read reviews from file
with open('train_pos.txt') as f:
    for line in f:
        review = line.strip()
        if len(review) > 0:
            reviews.append(review)

with open('train_neg.txt') as f:
    for line in f:
        review = line.strip()
        if len(review) > 0:
            reviews.append(review)

# Create features from text using a vectorizer. There are other vectorizers available, and lots of options you can set.
# This performs one step of computing word counts.
vectorizer = CountVectorizer(tokenizer=ng_tokenize, stop_words='english')
train_features = vectorizer.fit_transform([r for r in reviews])
text_features = vectorizer.transform([r for r in test])

# Fit a naive bayes model to the training data.
# This will train the model using the word counts we computed, and the existing classifications in the training set.
nb = MultinomialNB()
nb.fit(train_features, train_labels)

# Predict the scores.
for r in test:
    trainRes.append(1)
    for r in posReviews:
        trainRes.append(0)
    nb.predict(text_features)

# Now we can use the model to predict classifications for our test features.
predictions = nb.predict(text_features)
print(predictions)

# Calculate the area under the curve.
fpr, tpr, thresholds = metrics.roc_curve(actual, predictions, pos_label=1)
print("Multinomial naive bayes AUC: {}.".format(metrics.auc(fpr, tpr)))

```

**Classifier Comparison**  
Choosing the right estimator

**Using SVM**  
First Sample Kernel is extracted basis function i.e. exec(open('NaiveBayes.py')) where `naive` is a imported module

And then we convert each document to fit into any of to that vocabulary space in test features if a new word is appearing or that is called as out of vocabulary word that is ignored and only those counts are retained which are originally there in the vocabulary.

Once we have that we run this classifier and we also calculate the area under curve that gives us that basically give tells us about the false positive weight and the true positive weight, so for a binary classifier like this information helps us in identifying the performance of the classifier.

(Refer Slide Time: 18:14)

The screenshot shows a Jupyter Notebook interface with several code cells. The first cell contains code for Multinomial Naive Bayes:

```
# one we can use few more to prevent overfitting for our test features:  
# predictions = nb.predict(text_features)  
# Compute the error:  
# It is slightly different from our model because the intervals of this process work differently from our top  
# pr, tpr, thresholds = metrics.roc_curve(actual, predictions, pos_label=1)  
print("Multinomial naive bayes AUC: (%f)" %format(metrics.auc(pr, tpr)))
```

The output shows the AUC: 0.79508566612.

The second cell is titled "Using SVM" and contains code for an SVC with a radial basis function kernel:

```
In [1]: from sklearn.svm import SVC  
clf = SVC()  
clf.fit(text_features, train_labels)  
predictions = clf.predict(text_features)  
print(predictions)  
# Compute the error:  
# pr, tpr, thresholds = metrics.roc_curve(actual, predictions, pos_label=1)  
print("SVC Analysis AUC: (%f)" %format(metrics.auc(pr, tpr)))
```

The third cell is titled "Kernel as Linear Function" and contains code for an SVC with a linear kernel:

```
In [2]: SVC(C=1.0, cache_size=100, class_weight=None, coef0=0.0,  
       decision_function_shape='ovr', degree=1, gamma='auto', kernel='linear',  
       max_iter=-1, probability=False, random_state=None, shrinking=True,  
       tol=0.01, verbose=False)  
predictions = clf.predict(text_features)  
print(predictions)  
# Compute the error:  
# pr, tpr, thresholds = metrics.roc_curve(actual, predictions, pos_label=1)  
print("SVC Analysis AUC: (%f)" %format(metrics.auc(pr, tpr)))
```

The fourth cell is titled "Kernel as Sigmoid function i.e. tanh(gamma\*x + r) where r is specified by variable coeff"

```
In [3]: SVC(C=1.0, cache_size=100, class_weight=None, coef0=0.0,  
       decision_function_shape='ovr', degree=1, gamma='auto', kernel='sigmoid',  
       max_iter=-1, probability=False, random_state=None, shrinking=True,  
       tol=0.01, verbose=False)  
predictions = clf.predict(text_features)  
print(predictions)  
# Compute the error:  
# pr, tpr, thresholds = metrics.roc_curve(actual, predictions, pos_label=1)  
print("SVC Analysis AUC: (%f)" %format(metrics.auc(pr, tpr)))
```

(Refer Slide Time: 18:25)

The screenshot shows a Jupyter Notebook interface with several code cells. The first cell is titled "Using Random Forests" and contains code for a RandomForestClassifier with a gini criterion:

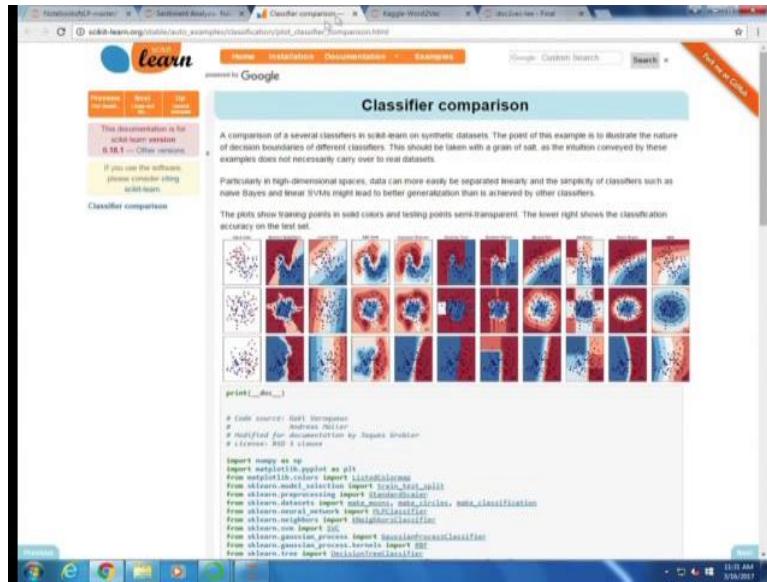
```
In [1]: from sklearn.ensemble import RandomForestClassifier  
RandomForestClassifier(n_estimators=10, criterion='gini', max_depth=None, min_samples_split=2,  
                       min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto',  
                       max_leaf_nodes=None, bootstrap=True, oob_score=False,  
                       n_jobs=1, random_state=None, verbose=0, warm_start=False, class_weight=None)  
predictions = clf.predict(text_features)  
print(predictions)  
# Compute the error:  
# pr, tpr, thresholds = metrics.roc_curve(actual, predictions, pos_label=1)  
print("SVC Analysis AUC: (%f)" %format(metrics.auc(pr, tpr)))
```

The second cell is titled "Maximum Trees" and contains code for a RandomForestClassifier with a gini criterion and a maximum depth of 2:

```
In [2]: RandomForestClassifier(n_estimators=10, criterion='gini', max_depth=2, min_samples_split=2,  
                           min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto',  
                           max_leaf_nodes=None, bootstrap=True, oob_score=False,  
                           n_jobs=1, random_state=None, verbose=0, warm_start=False, class_weight=None)  
predictions = clf.predict(text_features)  
print(predictions)  
# Compute the error:  
# pr, tpr, thresholds = metrics.roc_curve(actual, predictions, pos_label=1)  
print("SVC Analysis AUC: (%f)" %format(metrics.auc(pr, tpr)))
```

So, Scikit learn provides a host of classifiers it includes classifiers like SVM with multiple Kernel options or a simple classifiers like random forests. So, you can refer to any of the machine learning lectures to have an understanding of these classifiers for a quick check of what are the different classifiers that is provided.

(Refer Slide Time: 18:40)

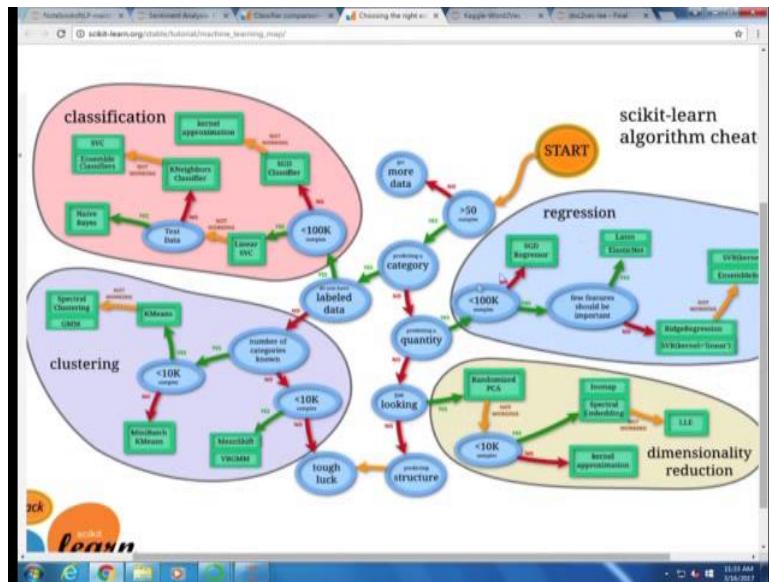


(Refer Slide Time: 18:40)



You can have a look into the scikit learns official documentation. So, the link is already provided here. So, here the classifier essentially provides different classifiers and they show on one data set how each of this classifier is performing you can have a look into this in addition.

(Refer Slide Time: 19:09)

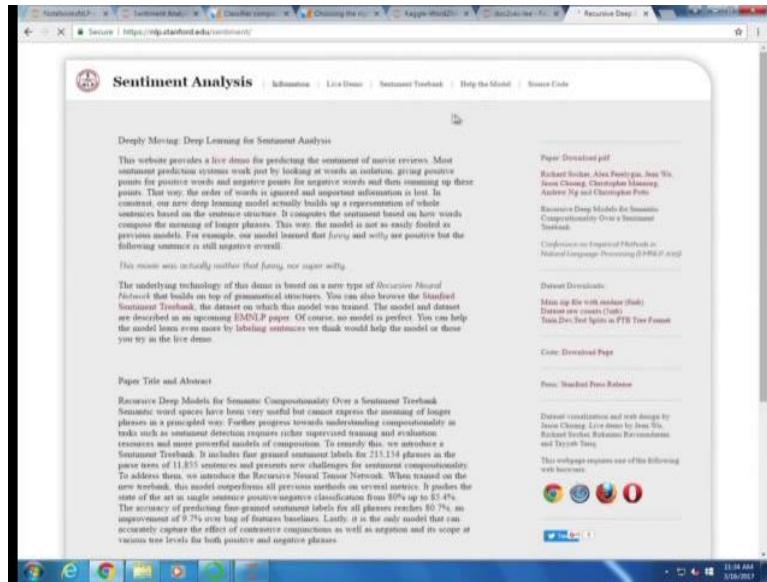


You can have a look into this quick scikit that will help you to find the relevant classifier or the estimator that you should be using. So, this scikit essentially provides all possible or it basically summarizes what are the different estimators that is available in scikit and you can see that when we do a supervise approach we should be either doing a classification or when we have where we have to classify or we have to predict a particular label.

In this case we were doing a classification where we predicting that whether something belongs to positive or negative. Suppose you wanted to predict the degree of positiveness or negativeness in a particular document you might have to model the same task as a regression task where it has a suite of library or suite of methods available here. In case of working with un supervised data like plastering or those kind of things that we have done in words disambiguation you might have to consider one of these clustering options or you can use the topic model that we have discussed in the previous tutorial.

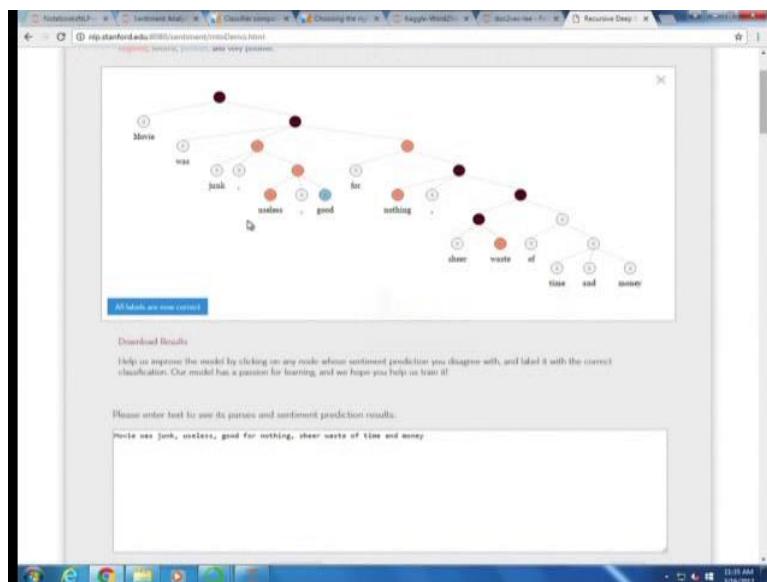
This essentially summarizes the class the supervised approaches for building a sentiment analysis tool.

(Refer Slide Time: 21:00)



You can also look into other standard sentiment analysis tools like the Stanford sentiment analyzer see if we give one of our sentences that we have used here for testing.

(Refer Slide Time: 21:37)



Here what we can find is that this again shows more or less negative sentiments to the classic the sentence. So, it basically shows each word to be either very negative or negative and very few words to have a positive impact. So, that will be all for this tutorial, please try out other classifiers as well.

Thank you.

**Natural Language Processing**  
Prof. Pawan Goyal  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 60**  
**Tutorial**

Hello everyone.

(Refer Slide Time: 00:21)



So, welcome to the tutorial session about word vectors or word embeddings. So, we will see how we can use word embeddings to find out relation between words or how we can find out again sentence representations or document representation using this word vectors. Fascinatedly aware that we use the distributional hypothesis that is a word can be defined by the words associated with it in its context. So, we use this notion to build up representations for word and then we (Refer Time: 00:58) similar words using this particular hypothesis. So, for this task what we will be doing is that, we will be again using the same documents the review documents that we have used in the previous tutorial.

(Refer Slide Time: 01:24)

```
# In [1]: import pandas as pd
# In [2]: df = pd.read_csv('polarity.csv')
# In [3]: df
```

Part 1 : simple bag of words model : Preprocessing the model

```
# In [1]: df = pd.read_csv('polarity.csv')
# In [2]: df
```

id	label	text
0	0	Macbeth is a rather example of an accident.
1	0	I like this film very much.
2	0	I want this with my friends up. So beauty of.
3	0	This book is quite expensice for the outside price.
4	0	China is a great place to go to - I enjoyed it.

Beautiful soup is a Python library for pulling data out of HTML and XML files. It works with Python's standard library for parsing documents, easy for manipulating, searching, and extracting the parts from it. Commonly, some programming errors or bugs of code.

So, I have collected all of them together into a single csv document here and we use a data since library pandas that helps us in speedy processing of these or more efficient processing of these documents.

(Refer Slide Time: 01:31)

```
# In [1]: df = pd.read_csv('polarity.csv')
# In [2]: df
```

Part 1 : simple bag of words model : Preprocessing the model

```
# In [1]: df = pd.read_csv('polarity.csv')
# In [2]: df
```

id	label	text
0	0	Macbeth is a rather example of an accident.
1	0	I like this film very much.
2	0	I want this with my friends up. So beauty of.
3	0	This book is quite expensice for the outside price.
4	0	China is a great place to go to - I enjoyed it.

Beautiful soup is a Python library for pulling data out of HTML and XML files. It works with Python's standard library for parsing documents, easy for manipulating, searching, and extracting the parts from it. Commonly, some programming errors or bugs of code.

```
# In [3]: df.info()
# In [4]: df
```

df looks like a pandas DataFrame of an document data containing around 5000 rows. Each row has two columns: 'label' and 'text'. The 'label' column contains binary values (0 or 1) indicating whether the document is positive (0) or negative (1). The 'text' column contains the raw text of each document. This is a common way to represent text data for machine learning models. The 'text' column is a string type, so we need to convert it to a numerical type for the model to work. We can use the 'CountVectorizer' class from scikit-learn to convert the text data into numerical features. This will create a sparse matrix where each row represents a document and each column represents a word in the vocabulary. The value at position (i,j) is the count of the j-th word in the i-th document.

```
# In [5]: df['text'].head(10)
```

[0] 'Macbeth is a rather example of an accident.'

[1] 'I like this film very much.'

[2] 'I want this with my friends up. So beauty of.'

[3] 'This book is quite expensice for the outside price.'

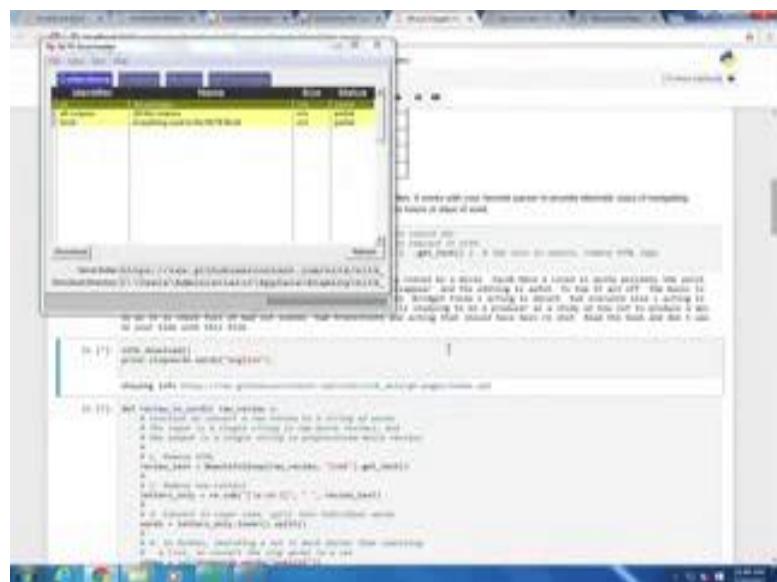
[4] 'China is a great place to go to - I enjoyed it.'

So, here we load all these documents into the data frame train. See if you find train essentially has about 5 3 8 7 documents, which are those documents, which are like compilation of both positive and negative documents. And you have previously seen those 5 samples from this document collection. So, as we were mentioning in the

previous tutorial that a lot of preprocessing needs to be done before using this particular document set.

For example, it has some extra html tags like br slash it has some other external markers. So, we will be removing the and we replace it with other normal characters. So, here this regular expression essentially looks for those entities which are in alpha and numeric. And now what we do is that we will also do stop word removal for this purpose we will be using the stop words that are stored inside nltk. See you can find nltk corpus we have imported the stop words. So, you might not have installed this by default in your systems.

(Refer Slide Time: 03:18)



So, what you can do is you can call the function nltk.download from the notebook itself. It will show a pop up box where you can go and look for stop words package, see you can just select on one of those packages say a stop words and click on the download button, we already have it downloaded. Similarly, we will be using the tokenizer which is the (Refer Time: 03:41) token tokenizer that also should be downloaded before this notebook has to be used.

So, we take the stop words. So, we I will be commenting it out yeah and we find these are the stop words. Once we have this, what we do is that we convert each review and we extract each individual word from those reviews. And in that process we do the case folding and we remove the stop words and all other necessary preprocessing. So, we will

be doing it for all of those sentences once we have it.

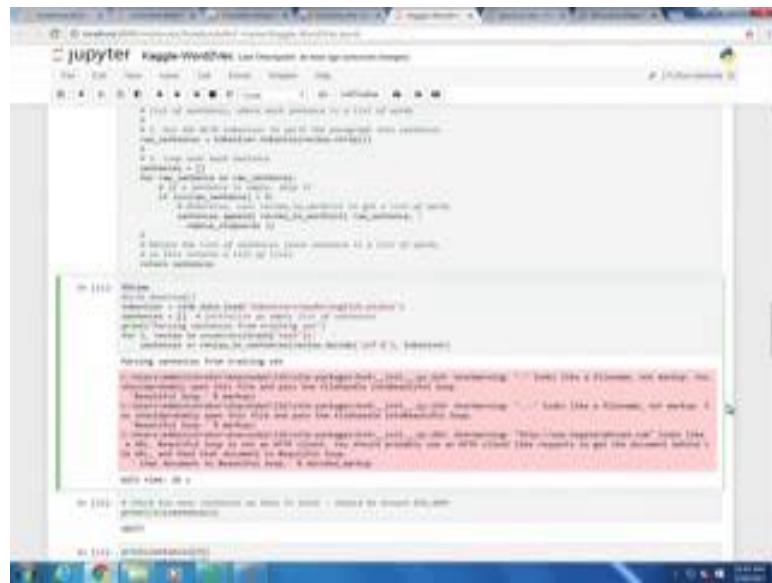
(Refer Slide Time: 04:30)



We need to have a suitable vector representation or a list representation initially and then convert it into a vector using the word embedding.

Yes, now we will be using the word to an embedding method. So, I assume you remember the word2vec embedding method that is taught in the class. Otherwise I recommend you to go through the lectures and see how exactly word2vec functions.

(Refer Slide Time: 05:05)



Unfortunately, we have genism package that provides us the word to a pre implemented in in python. So, once we convert this sentence, we will be training the model. So, in Jupiter notebook you can use magic functions or magic keywords, where if you use something like time, it will tell how much time if it requires to execute a particular function.

So, from all the 5000 documents we have extracted about 58,000 different sentences. So, we can have a look into how the sentence looks like. So, the first sentence looks like this movie is a perfect example of an excellent book getting ruined by a movie. So, there are other reviews as well.

(Refer Slide Time: 06:18)



Now, we have to convert each of this into a word vector. So, for each of this word we have to get an efficient vectorial representation or a vectorial representation.

So, what we do is that we have to set some parameters before we use this model.

(Refer Slide Time: 06:38)



So, one thing is that we have to find how many or what is the dimension that the vector should have once word to a glance the vectors. So, we expect it to be somewhere around 300. So, this is often set empirically after trying out various dimension sizes. So, here output vectors all are for each word will be of size 300 now minimum words count. So, minimum words count is basically the number of times a word has to appear in the corpus to be considered for the vector formation, may be for this practical purpose I for this tutorial purpose I will be keeping it a bit lower.

So, this is a programming convenience is that what are the number of parallel threads to run this makes you a job faster, but it depends on the ability of your system. Next comes context window size. So, context window size essentially looks for given a sentence how many near nearby words have to be considered. So, if I consider the word example how many words left to this given word and right to the given word should be considered while forming the vector for this word. So, as we know distribution semantics depends on the word and it is nearby context and we here set a cap on, or a limit on the number of word that can appear in it is context. Whatever appears we need this count will be ignored by the model when we prepare the vector for each word.

So, though we have about 58 sentences in the interest of time, I will be using a less number of sentences here. So, we use the word to work model. This word to work model is defined in genism. So, that we can see that from genism I am importing word2vec.

(Refer Slide Time: 08:49)

```
jupyter Notebook
```

```
In [1]:
```

```
w2v = Word2Vec(min_count=1, size=300)
```

```
In [2]:
```

```
w2v.train(sentences)
```

```
Training word2vec on 100000 sentences took 3.000 seconds
```

So, once we have word2vec we keep it size to be 300, which is number of features and we keep then min word count the number of words that should appear. I mean the number of times it should appear before it appears in a corpus because it needs to be considered.

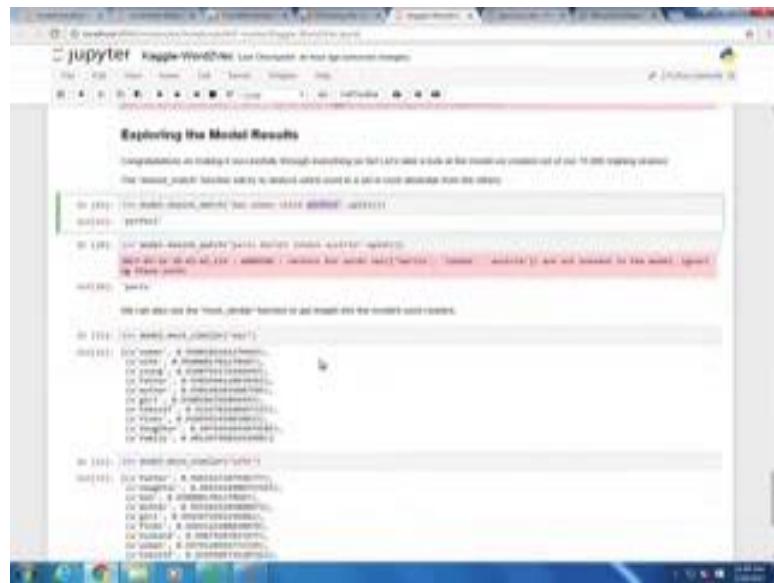
So, yeah when it comes ignore all those words with the total frequency lower than that particular count. So, you can use the shift tab keys in your keyboard to see the definition for each of these functions. Now there is this parameter sg equal to 0. So, if you remember there were 2 ways of using word2vec one was the skip gram model and the other was the cbow model. So, when we keep sg equal to 0 cbow is used, but if we use sg equal to one the system will or the model will use skip gram model. So, here by default we are keeping sg equal to 0. See we will train the model and let us see how much time it takes. So, in around 3 seconds we have our model trained.

So, let us try to increase the count of sentences and see how much more time it takes. See it takes almost double the size. So, there are some other arguments that you need to go through for example, there we give a down sampling as the argument to the as the value to the argument sample. So, it essentially says that if you have a certain number of words that is much more than what we require they are randomly sampled. Similarly, here negative. So, when we use this function with this argument negative it basically takes care of the negative sampling that word to efficiently uses.

So, by default it looks for 5 negative samples. It can be set to varying values that will

naturally effect the quality of the vectors that are generated.

(Refer Slide Time: 11:39)



So, now what we do is that we, once we have the model where we have a vector for each of those words which are there in 50,000 sentences what we do is that we give a new sentence and see which of these words are a most dissimilar to that in the model. So, we can find that kitchen is somehow not present in the word. So, it is ignored. So, amongst man woman and child, here child is the most dissimilar word from the other 2 words that is man and woman. For example, let us see some other word.

Say let us take one of those words like perfect and see is this change the result. So, now, amongst the 4 words man women child and perfect, you find that perfect is the most dissimilar to the other 3. Now we can also have this function most similar which basically tells given a word what are the other similar words in the entire corpus. So, we can find the word man has other similar words which are women killer father etcetera with it is corresponding similarity score. The similarity score that typically used is causing similarity though other similarity functions can also be efficiently used.

So, now if we find the word wife, the similar word that appears with the son father brother girlfriend with it is different similarity scores. So, though we cannot tell the exact relation between the words that is in the list, what we can find here is that the words that appear similar to this word are in reality similar to the given word. So, if we give the word aw full we find other similar words that can be used in same context as the word

awful can be used we can find terrible horrible boring dull etcetera. So, this is essentially how we can use word2vec to find similar words.

(Refer Slide Time: 14:13)



So, you can have a look in to the Kaggle challenges where they have different competitions where we can use the word vectors. So, we have taken this tutorial from the Kaggle tutorial or the Kaggle competition called when a bag of words meet bag of popcorn, where you are supposed to build a task where you basically use the review dataset to build word vectors and then use them for the sentiment analysis task that we have previously seen.

(Refer Slide Time: 14:47)



So, they have different parts to this particular tutorial. It will be advisable to go through each of this. In addition I will be just referring to the doc2vec which is an improvement over our word2vec.

(Refer Slide Time: 15:04)

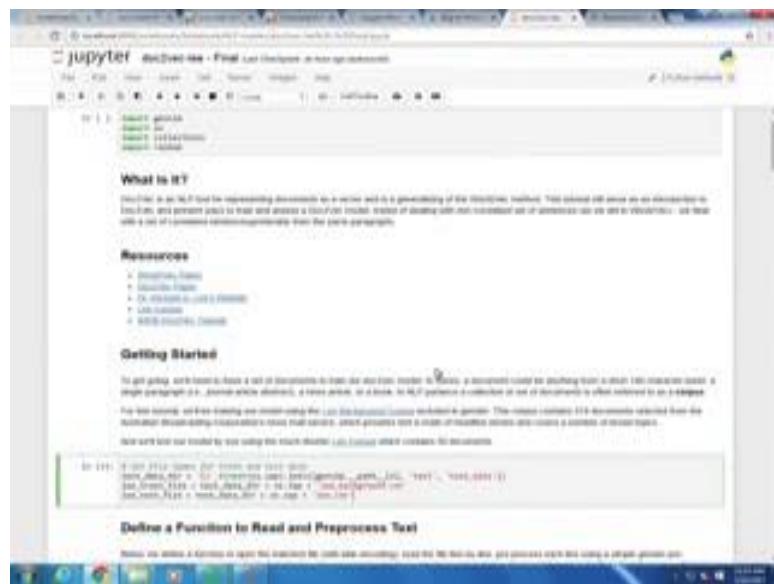


Or which is basically it cannot be set as an improvement, but it is a method to create vector representation for the document. See we have seen in the topic modeling tutorial that we had representation vector representation for each document in this particular tutorial, we were not using any document representation we were using a vector for each

individual words.

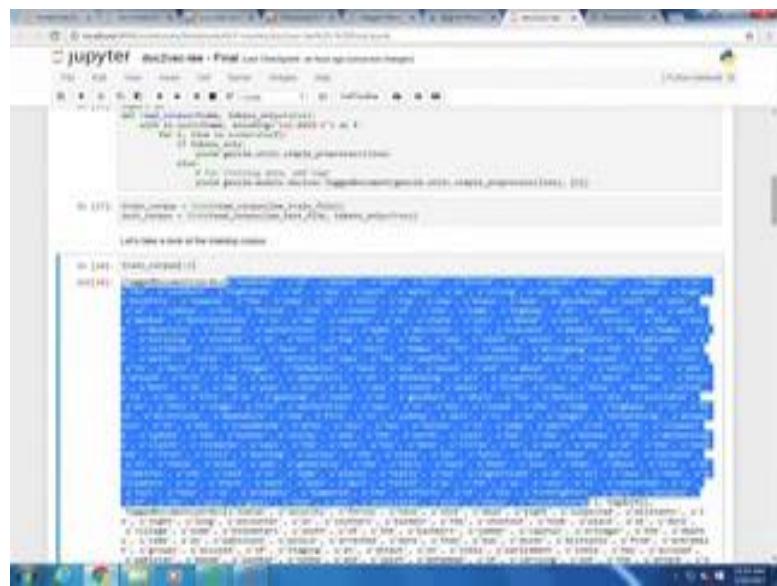
Now, how can we use this word2vec vectors to arrive at a document vector representation and we have this doc2vec method. So, you can look in to the doc2vec paper here which is linked to this particular notebook. Here we will be using the default tutorial provided in genism.

(Refer Slide Time: 16:01)



So, genism already has a corpus and it already has the doc2vec implemented in this model. So, we can just import the corpus that is stored in genism and we can find that they provided tagged corpus.

(Refer Slide Time: 16:20)

A screenshot of a Jupyter Notebook interface. The code cell contains the following Python code:

```
from gensim.models import Doc2Vec
from gensim.test.utils import get_tmpfile
from gensim.models.doc2vec import TaggedDocument
from collections import defaultdict
import numpy as np
import pandas as pd
import os
import re
import string
import nltk
nltk.download('punkt')
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
```

The output cell shows a large matrix of word embeddings, represented by a grid of blue numbers. The first few rows of the matrix are:

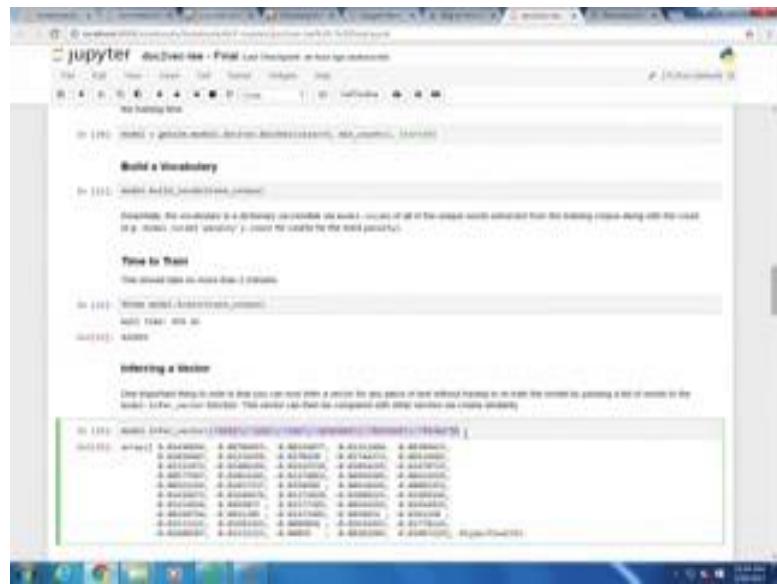
```
[ 0.00000000e+00  0.00000000e+00 ...,  0.00000000e+00  0.00000000e+00]
```

So, here is the corpus with individual words that forms the document and its corresponding tag is given.

Similarly, there is another document with its tag given as one now we have a test corpus. So, the test corpus the tag is not provided. Now we instantiate the doc2vec model, so as in the word2vec you can find different arguments that you can give to the doc2vec so here size is the dimensional vector of the feature vector. So, now, each document will be represented as a vector with the given size and you can look into other arguments as well.

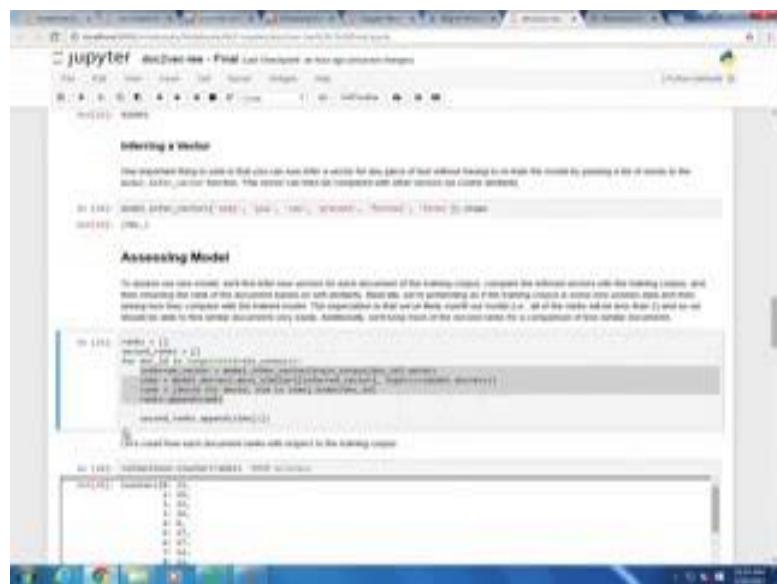
Now, once we have this model we build the vocabulary and we use it to train the corpus.

(Refer Slide Time: 17:24)



Now, with this model when you have this vector, that has the following words only you can prevent forest fires, we get a representation for that particular document. We can see what is its size. So, we can find that the size of this particular vector is of 50.

(Refer Slide Time: 17:51)



Now, once we have this vector for each of the document in our training corpus what we do is, that we take a particular document and we find its similarities with other vectors or other documents and we then form a rank for each of them. Once we have this once we have this computer the similarities. What we have done is that we have taken the vectors

for each different document and we have compared amongst themselves and we find the similarity between those documents.

(Refer Slide Time: 18:28)



So, what do you find here is basically the similarity of a given document and it is rank. So, lower the number it is better and we can find that lot of them have very high similarity and they have they share the same value hence are ranked at 1. So, let us see which are those documents.

(Refer Slide Time: 18:54)



So, we were comparing this document and we find that the most similar and dissimilar

documents are these documents. So, we can find the document 299 has the highest similarity or one of those with the highest similarities 83 and you can find you can have a look in to these 2 documents.

(Refer Slide Time: 19:14)



So, essentially if you see there are the contents in the both the document, basically talks about very similar manner very similar issues and hence they have similar words in the context. And thereby the model is able to find the document similarity. Here we find another document which is at the median. So, we can find that the similarities called between those at the highly ranked and the median are not much different. Hence still we can find the difference in the words that are used in context of all these documents.

(Refer Slide Time: 19:56)



And the least similar document given here is this.

(Refer Slide Time: 20:00)



So, all these documents are essentially coming from a single work or a single book and these are different paragraphs of the same article. So, they are more or less similar, still they have difference relative, differences within their similarity.

So, the doc2vec essentially helps in handling the documents similarity between documents in a given corpus. So, this will be this is all that we will be discussing regarding distribution similarity, or distribution semantics especially using word vectors.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 61**  
**Sentiment Analysis – Introduction**

Welcome back for week-12 of this course. So, from last two weeks, we have been talking about lot of different applications in NLP and what kind of methods you can use for solving those applications. And we have discussed applications about text classification, text summarization, entity linking and functionality extraction. There are many other applications where of NLP that you can think of. And once you have learnt all the basic concepts, so you have to just see how what kind of algorithms you can apply for newer applications.

So, this week we will focus on one of the very, very interesting application that is of sentiment analysis and we will cover that from many different angles. So, sentiment analysis is a field that has become very, very popular in the last decade; and because of the social media coming in and lot of opinions and comments that people are putting online. So, it becomes important to understand what are peoples sentiment towards a particular product; a particular movie, a particular candidate in politics and so on. And can one do that by simply analyzing, the sort of text data that I find online and also on various different resources.

So, can I use that to do some sort of sentiment analysis. And this can later use for various predictions that how people sentiments will move in future and may be what will the market of particular product or a movie and so on. You can also do some sort of comparisons between products, people and so on. So, we will see what this field of sentiment analysis enters? And what are some of the things that will help you on taking up any new challenging problems sentiment analysis? So, this is what can be some sort of introduction to sentiment analysis that you have some sort of movie reviews and each reviews either saying ok this movie is very positive talking positively about the movie or the negatively about the movie.

(Refer Slide Time: 02:28)

*Example: Positive or negative movie review?*

- unbelievably disappointing
- Full of zany characters and richly applied satire, and some great plot twists
- this is the greatest screwball comedy ever filmed
- It was pathetic. The worst part about it was the boxing scenes.

Pawan Goyal (IIT Kharagpur) Sentiment Analysis - Introduction Week 12, Lecture 1 2 / 16

So, let us see some of the movie reviews here. So, it says unbelievably disappointing, and you may say it is this is negative review. Another way is ones is full of zany characters and richly applied satire, and some great plot twist, and you will say this is a positive review. This is the greatest screwball comedy ever filmed, you will say again as a positive review. It was pathetic. So, you see this becomes a negative review. So, you are having lot of different reviews and you are seeing which one is positive, which one is negative and so on.

(Refer Slide Time: 03:04)

*Where is Sentiment Analysis Used?*

*Movie* Is this review positive or negative?  
*Products* What do people think about the new iPhone?  
*Public Sentiment* How is consumer confidence? Is despair increasing?  
*Politics* What do people think about this candidate or issue?  
*Prediction* Predict election outcomes or marked trends from sentiment

Pawan Goyal (IIT Kharagpur) Sentiment Analysis - Introduction Week 12, Lecture 1 3 / 16

What are the different places where you can use sentiment analysis in general? So, we took

some examples from the movie domain. So, from the way people write reviews about the movie, you can say that whether the movie was the reviews are positive or negative, but you can talk in general about many different things. For example, you can say in terms of products. So, when you buy products online, there is also a field where you can put your reviews, you can put your reviews about the product.

And there are again one can capture whether about the product are using something positive or using something negative and so on. You can also go for the down saying this product has multiple aspects. So, this product like camera, you can talk about whether it is very heavy camera, whether or how good image, what is the image quality and so on. So, you can talk different aspects about the product; and each aspect you can say whether the opinion of the sentiment of the people is positive or negative.

Then you can enjoy talk about the public sentiment that is it can be over a particular brand name. So, on that brand, what is the consumer's confidence, what is the consumers reactions, it can be in general about the policies that government is making that what is the confidence of the people in that policy and what is their sentiment and so on. You can do it for the politics that is whenever there are some elections, some candidates are there. So, what are the people's opinions and sentiments about different sort of different candidates? And this is again a very, very interesting field where you can say ok is it just positive negative or you can you can go further say ok. What positive things are people talking about a particular and candidate what negative things people are talking about this candidate and you can also do some lot of summarization on top of that. So, there is a lot of different things that you can do about this field of sentiment analysis.

And by doing all that you can also do some sort of prediction that is once I know the peoples sentiments about different political parties or candidates, can I predict the outcome of the election in advance. And you will see in the recent years, this has being done in a lot and lot to by many different sorts of researchers and companies. So, they are trying to predict the outcomes of elections, whether it is US election, whether it is elections in India and so on.

Can you predict the market trends from sentiments? So, once you know the sentiment of the people, can you predict which sort of product will have a high market; you can also talk about the stock exchange about what stocks will go up, what stocks will go down and so on. So, there are lots of different interesting things that can be done using sentiment analysis.

(Refer Slide Time: 05:57)

The slide has a blue header bar with the title 'Where is Sentiment Analysis Used?'. The main content area is white with a black border. At the bottom, there is a dark footer bar containing the text 'Ptwan Goyal (IIT Kharagpur)', 'Sentiment Analysis - Introduction', 'Week 12, Lecture 1', and '4 / 16'.

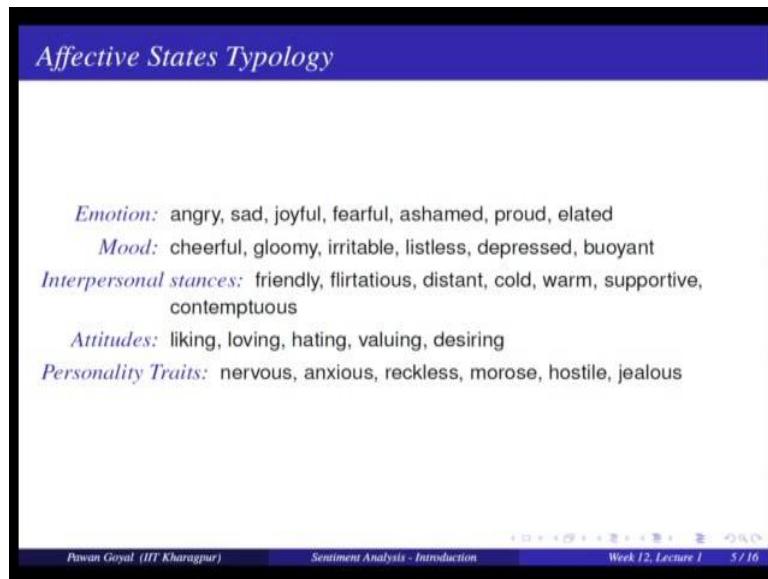
- Frustration of callers to a help line
- Stress in drivers or pilots
- Depression and other medical conditions from social media
- Confusion in students talking to e-tutors

Then there are other things like your help line and you want to see whether the scholars are somehow distressed or the frustrated and so on by doing sentiment analysis over their speeches. Then from the drivers and pilots, she want to find out the stress labels there that again it become important to avoid various accidents; so there also sentiment analysis used. Although their modality might be different, you might not be able to capture it by using simply the text data you might have look at their speech or some other sort of facial expression and all. So, in this course, and in this field we will only talk about taking it from the text data.

Then so that is why the other two kinds of applications become interesting that is the way people are using social media, so by the way they are writing over social media by the way they are making the connection with people, can we tell something about their health conditions like are they depressed, and they are works in recent times that are explore this area that is. Can you tell that by the way the tweeting behavior of a person, whether he is depressed, whether he is alcoholic and so on? And then when people are using various e-learning tools they are talking to e-tutors are they understanding the topic or are they having some confusions.

So, these sentiments can also be explored by using sentiment analysis. And you can think of hundred different scenarios, where this sentiment analysis will be used.

(Refer Slide Time: 07:51)



So, before going on to the how we do sentiment analysis let us discuss, what all comes in this broad field of sentiment, opinions, subjectivity what all can you think about. So, this is by this affective states typology. So, what are all the different things you can talk about this broad field? So, you can see talk about emotions right, emotions of something different. So, emotions will be are you angry, are you sad, joyful, fearful, ashamed, proud, elated all these are emotions.

Then you can top talk about the moods. So, whether the person is cheerful, he is gloomy, is it irritable, listless, depressed, buoyant. So, all these are different moods of people. Then one can talk about the interpersonal stances that are whether he is friendly, whether he is warm, supportive and so on. You can talk about attitudes – liking, loving, hating, valuing, desiring. And you can talk about the personality traits like being nervous, anxious, and reckless. And you will seen different, different applications, you might want to talk about different sort of states somewhere you want to talk about the emotions, somewhere only the interpersonal stances, we are talking about a group how people are behaving in a group.

So, if you talk about the social media and how people are reacting toward ho how peoples comments are, you will see we are mostly talking about the attitudes of the people that is are they liking a product or they loving it or they hating it same about the movie or the candidate. So, when we talk about sentiment analysis, we mostly focus on these finding of this attitudes of people towards certain object or a person.

(Refer Slide Time: 09:28)

**Sentiment Analysis**

*Sentiment Analysis is the detection of attitudes  
enduring, affectively colored beliefs, dispositions towards objects or persons*

**The complete task**

- Holder (source) of attitude
- Target (aspect) of attitude
- Type of attitude
  - From a set of types: *like, love, hate, value, desire*
  - Or simple weighted polarity: *positive, negative, neutral, together with strength*
- Text containing the attitude

Pawan Goyal (IIT Kharagpur)      Sentiment Analysis - Introduction      Week 12, Lecture 1      7/16

So, this is some sort of definition we can give that sentiment analysis, the detection of attitudes that is some sort of enduring our some sort of effectively colored beliefs, towards objects or person. So, now in general sentiment analysis can be a very, very complex task also. So, because when you are talking about sentiment analysis, there are many, many different things in picture. So, you are talking about the person who is the person who is holding the attitude, who is the person, who is liking it that is the holder of the attitude; and liking what, what is the target of this attitude. So, these are two important things who is the holder who is the target.

Then you can talk about what is the type of attitude. So, again here you can choose from this wide range, you can say I have to choose one of the like, love, hate, value, desire and so on. Or simply you can say what is the polarity is it positive, is it negative, is it neutral or you can give some sort of strength is positive with that much of strength, negative with this much of strength and so on. So, again you see there is a lot of flexibility in how you can define your task. And you can some might be more complex than the other.

And you might also want to find out what is the text in this whole whatever attendance you have taken that contains that attitude. What part of text contains that attitude among this whole sentence or paragraph? So, this might be the complete task. Now, you might just want to use one of this task and not the other, one of the particular things like I want to just say where this whole sentence is positive or negative I do not worry what part is positive or

negative. Or I want to say who is having the opinion and towards whom and that is all I want to do not want to go further into detail. So, you can choose which of what part of this whole task, you want to compute to for your particular application.

(Refer Slide Time: 11:29)

The slide has a blue header bar with the text "Sentiment Analysis". Below it is a white content area. At the top of the content area, there is a purple horizontal bar with the text "Simplest Task" in white. Underneath this bar, the text "Is the attitude of this text positive or negative?" is displayed. Below this is another purple horizontal bar with the text "More complex" in white. Underneath this bar, the text "Rank the attitudes of this text from 1 to 5" is displayed. Below this is a third purple horizontal bar with the text "Advanced" in white. Underneath this bar, the text "Detect the target, source, or complex attitude types" is displayed. At the bottom of the slide, there is a dark blue footer bar with the text "Piwan Goyal (IIT Kharagpur)" on the left, "Sentiment Analysis - Introduction" in the center, and "Week 12; Lecture 1 8 / 16" on the right. There are also small icons for navigation and search.

So, simplest task would be you are given a text and is the attitude positive or negative that is the simplest task. And you will see most of the works are done for this simple task. So, and if you want to make it more complex, you will say ok, all these attitudes are there can I give them some sort of score. So, I score them in the range of 1 to 5, 1 might mean very negative and 5 might mean very positive; it can be from minus 5 to 5.

So, they are different scales on which you can give the ratings 1 to 5, 1 to 10 and so on. You will see when you give rating to movies or the products on different sites; you have ratings from 1 to 5 or 1 to 10 and so on. So, again you might want to predict the ratings on that that is skilled from the text and yes, then it can be more advanced where you want to detect who everything target, source, and the complex attitude types. So, you can see that definition can vary. It can be simple, more complex or advanced sentiment analysis.

(Refer Slide Time: 12:37)

The screenshot shows a slide titled "Sentiment Analysis in Movie Reviews". A sub-section titled "Polarity detection" asks, "Is an IMDB movie review positive or negative?" Below this, there are two reviews side-by-side. The left review, marked with a green checkmark, discusses "Star Wars" and is labeled as positive. The right review, marked with a red X, discusses "October Sky" and is labeled as negative.

**Positive Review (Left):**

when \_star wars\_ came out some twenty years ago , the image of traveling throughout the stars has become a commonplace image . [...] when han solo goes light speed , the stars change to bright lines , going towards the viewer in lines that converge at an invisible point . cool .

\_october sky\_ offers a much simpler image—that of a single white dot , traveling horizontally across the night sky . [ . . . ]

**Negative Review (Right):**

" snake eyes " is the most aggravating kind of movie : the kind that shows so much potential then becomes unbelievably disappointing . it's not just because this is a brian depalma film , and since he's a great director and one who's films are always greeted with at least some fanfare . and it's not even because this was a film starring nicolas cage and since he gives a brauvara performance , this film is hardly worth his talents .

Pawan Goyal (IIT Kharagpur)      Sentiment Analysis - Introduction      Week 12, Lecture 1      9 / 16

So, let us see some example. So, you are given IMDB, you are having two different reviews and you want to find out whether they are positive or negative. So, here are two reviews. So, on the left hand side, so you have about when a star wars came out some twenty years ago, the image of traveling throughout the stars has become a image and then you find something like cool. And this is a movie October sky it offers a much simple simpler image white dot traveling horizontally across the night sky.

And this overall look gives a positive picture. We have the word like cool here and this gives a positive picture. And then if you see the one on the right hand side, and you will see immediately there is a word like this is unbelievably disappointing, and this film is hardly worth his talents and so on and you will see this become a negative review. So, from this whole text, you want to find out whether this is positive review or negative review.

(Refer Slide Time: 13:38)

*Baseline Algorithm*

- Tokenization
- Feature Extraction
- Classification using different classifiers
  - ▶ Naïve Bayes
  - ▶ MaxEnt
  - ▶ SVM

Pawan Goyal (IIT Kharagpur)      Sentiment Analysis - Introduction      Week 12, Lecture 1

So, now what are the some simple algorithm that you can apply. So, if you think about it for a while you can say this looks like a classification problem. So, we did classification just in the last week. So, you are having different text, and you want to classify them into either positive class or negative class; whether this is positive or negative. And we know we can use a lot of different models of classification, I can use SVMS, I can use logistic regression, I can use Naïve Bayes model that we talked about. So, I can use any of these models.

But before doing that I will probably have to see given the text do some sort of tokenization, find out what are the words here, and I extract what are the important features that I can use for my classification tasks. So, this has to be done before running any of this algorithms. So, what is important is that can we do something that is specific to this task without about classification problem in general. And we know how to do all these for any given text. We will need to have some labels and we will learn Naïve Bayes classifier. But what is important for this task and that is something that you have to keep in mind when we are dealing with any application. So, there are some default set of methods that you can use for that application, but you have to see is the something is specific out of this application that you should use.

So, let us talk about tokenization right; we discussed tokenization in the very first week of this course. So, I need to find out what are the different words here. And there I do some sort of preprocessing. So, what are some sort of preprocessing. So, I do lower casing and I correct

the spellings and so on. And if there are words that auto vocabulary and might also remove them. So, there are many things I can do. But when we are talking about sentiment analysis do you needs do something different, nothing about the tweets

(Refer Slide Time: 15:50)

### Tokenization Issues

- Capitalization - preserve for word in all caps
- Word lengthening
- Handling emoticons

```

[<>]?                                # optional hat/brow
[+;=8]                                 # eyes
[\-\o\*\\']?                            # optional nose
[\\]\(\[dDpP/\+\])\{@\\}\\|               # mouth
|                                         ##### reverse orientation
[\\]\(\[dDpP/\+\])\{@\\}\\|               # mouth
[\-\o\*\\']?                            # optional nose
[+;=8]                                 # eyes
[<>]?                                # optional hat/brow

```

- Handling negation
  - ▶ I **didn't** like this movie
  - ▶ I really like this movie

Add ***NOT\_*** to every word between negation and following punctuation

- ▶ *didn't like this movie, but I ...*

Piwan Goyal (IIT Kharagpur)      Sentiment Analysis - Introduction      Week 12, Lecture 1      11 / 16

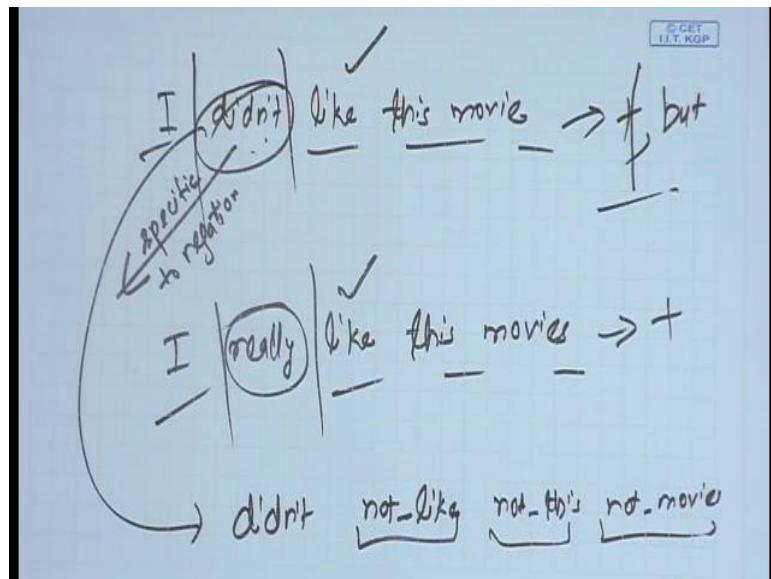
So, let us see what would a capitalization. So, suppose you are having a comment where the person is writing in all caps. So, what can you say immediately about that thing. So, whenever you are writing in all caps, in general that might indicate some sort of sentiment. So, you write cool in all caps, probably you are very, very happy about it or you are liking this. So, you want to preserve that this word occurred in all caps, you want to preserve that information, and you did not want to leave loose that information by doing lower casing. Then again you might have some lengthening of the words right. So, you are writing cool and you are writing five or six o in between c and l. So, you want to preserve that you are doing a lengthening here and this lengthening can again mean some sort of sentiment.

Then one interesting thing that comes is the emoticons that we never worried about, but in sentiment analysis emoticons are a very good indicators of the sentiment of people emoticons can be either in comments when people are chatting to each other when they are tweeting. So, are they are they some easy way of caption the emoticons, so yes, they all come from some sort of regular expressions. So, if you can build some regular expressions and that are actually available with us

So, here are some example of regular expressions you can use and you can detect this

contains a an eyes, optional nose, mouth, reverse orientation and so on, and each of this emoticon has some sort of sentiment. So, you can associate sentiments with each of these emoticons. Then another important thing is handling negation in sentiment analysis.

(Refer Slide Time: 17:54)



So, let us take these two examples. I did not like this movie versus I really like this movie. So, what will happen when you are doing the applying the standard Naïve Bayes model, so you will have this I did not like this movie, and then you have I really like this movie. So, what will happen in these sentences everything else is same except one word, and may be it can happen that because you are having a like you get a positive sentiment in both, yes, if you are doing a simple method you are applying a simple method by tokenizing taking each word and running a classifier.

So, question is can we do something specific to this negation. And while there are many different ways people handle that one interesting idea is you apply not to all the words that are following this. So, you say ok convert that to I have did not then not like. So, it becomes a different token right like is one token and not like is another token, not this and not movie something like that. So, you append not to all the words that are following it. And you do that until there is punctuation. So, like here it can happen that after punctuation I am starting something. But so you do not want to carry forward this negation to after punctuation see you want to stop it here at punctuation. So, this is one way you can handle this.

So, what I want to convey is that although there are some standard things that we have talked

about in this course, you do this lower case, you do this stemming, you do this and that, for a particular application you might have to deal with this differently. See you want to preserve here what are the upper cases so and what are the lengthening and what are the for negation you have to do something different find out the emoticons take them as an important features and so on. So, you have to do something specific for this application. See, you have to look like this did not, not like, not this, not movie.

(Refer Slide Time: 20:05)

*Naïve Bayes: Reminder*

$$c_{NB} = \arg \max_{c_j \in C} P(c_j) \prod_{x_i} P(x_i|c_j)$$

$$\hat{P}(c_j) = \frac{doc - count(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i|c_j) = \frac{count(w_i, c_j) + 1}{(\sum_{w \in V} (count(w, c_j)) + |V|)}$$

Piwan Goyal (IIT Kharagpur)      Sentiment Analysis - Introduction      Week 12; Lecture 1

$$c_{NB} = \arg \max_{c_j} P(c_j) \prod_{x_i} P(x_i|c_j)$$

$$\hat{P}(c_i) = (doc - count(C = c_i))/N_{doc}$$

$$\hat{P}(w_i|c_j) = (count(w_i, c_j) + 1)/(\sum_{w \in V} (count(w, c_j)) + |V|)$$

So, Naïve Bayes you remember we did in the last week itself. So, we will build a classifier suppose our classes may be two or three positive negative and neutral and this is how you build a classifier. And you can find the probability of each class by from your training data you can find the probability of each feature in the class features can be ensemble words or something else again from trained data using this maximum likelihood estimate with add one smoothly.

(Refer Slide Time: 20:37)

The slide has a blue header bar with the title "Boolean Multinomial Naïve Bayes". Below the header, there is a list of two bullet points:

- First remove all duplicate words from a test document  $d$
- Then compute NB using the same equation

Below the list is a mathematical formula:

$$c_{NB} = \arg \max_{c_j \in C} P(c_j) \prod_{x_i} P(x_i | c_j)$$

At the bottom of the slide, there is a footer bar with the following text: "Piwan Goyal (IIT Kharagpur)" on the left, "Sentiment Analysis - Introduction" in the center, "Week 12, Lecture 1" on the right, and "13 / 16" at the far right.

You can remove the duplicate words and you can use Naive-Bayes model to obtain what is the class, optimal class as per Naive-Bayes algorithm.

(Refer Slide Time: 20:51)

The slide has a blue header bar with the title "A piece of cake?". Below the header, there is a question: "Is a given review on a known topic positive or negative?"

Inside a red-bordered box, there is a quote:

"It may be a bit early to make such judgments, but Battlefield Earth may well turn out to be the worst movie of this century." (Elvis Mitchell, May 12, 2000)

Below the quote, there is a blue text: "don't we just need to look for "worst", "best", "love", "hate", etc.?"

On the right side of the slide, there is a circular profile picture of a man with glasses and a light blue shirt.

At the bottom of the slide, there is a footer bar with the following text: "Piwan Goyal (IIT Kharagpur)" on the left, "Sentiment Analysis - Introduction" in the center, "Week 12, Lecture 1" on the right, and "13 / 16" at the far right.

So, this is a simple way, but what else people have tried and mind you this symbol Naive-Bayes works sometime, but it does not work with the very good precision, and there are many issues why it does not work. So, we will see some of those. So, as such if you talk about sentiment analysis, so it looks a simple problem right. I just have to so let us take this particular review it may be bit early to make such judgments, but Battlefield Earth may well

turn out to be the worst movie of this century.

So, I read this review. And does that look very easy to find sentiment you say ok, yes, I see the worst movie and the worst is sufficient to tell me that this is a negative review. So, I can immediately assign it a polarity of negative, it looks easy. So is not it like I have to just see these words worst, best, love, hate in the reviews and then I can assign a sentiment is that as is.

(Refer Slide Time: 22:01)

In a small scale experiment (Pang et al., 2002)		
	Proposed word lists	Accuracy
<b>Human 1</b>	<b>Positive:</b> dazzling, brilliant, phenomenal, excellent, fantastic <b>Negative:</b> suck, terrible, awful, unwatchable, hideous	58%
<b>Human 2</b>	<b>Positive:</b> gripping, mesmerizing, riveting, spectacular, cool, awesome, thrilling, badass, excellent, moving, exciting <b>Negative:</b> bad, clichéd, sucks, boring, stupid, slow	64%
<b>Statistics-based</b>	<b>Positive:</b> love, wonderful, best, great, superb, beautiful, still <b>Negative:</b> bad, worst, stupid, waste, boring, ?, !	69%



Piwan Goyal (IIT Kharagpur) Sentiment Analysis - Introduction Week 12, Lecture 1

So, it turns out it is not that easy. So, then experiment was reported in Pang et al 2002. So, what they did, so they took a corpus of again I think some reviews, and they asked the people before showing the corpus, if they can come up with a list of positive words and negative words. And I will label a review as positive, if it contains a word positive; negative if it contains positive negative or the majority of both. And people then came up with different, different lists.

So, for example, so human one came up with this list like dazzling, brilliant, phenomenal, excellent, fantastic and so on; and negative suck, terrible, awful, unwatchable, hideous. Human two came up with this list of gripping, mesmerizing, riveting, spectacular, cool; and negative as bad, clichéd, sucks, boring, stupid so and on and so forth. So, they came up with different, different words in positive and negative class, but when they tried to see what will be the accuracy of using this word list that are created by humans on this task, they found that human 1 had an accuracy of 58 percent and human two had an accuracy of 64 percent.

On the other hand, if we try to obtain this from statistics, you get words like love, wonderful, best, great, superb, beautiful, words like a still also come up, because this is obtained using statistics; and negative we have words like bad, worst, stupid, waste, boring and some question mark exclamation etcetera. And this is still does better than the other two. So, this gives a performance of 69 percent than the other two. So, you see simply by using the words may not be sufficient, and why they that might be the case. So, why using these positive, negative words are not sufficient. So, mind you they are very important for sentiment analysis.

So, next lecture, we will complete devote on that that, what are different sort of lexicons or dictionaries about sentiment that you can use for this task, but what is the problem one problem with them.

(Refer Slide Time: 24:07)

Why can't we just look for words like "great" and "terrible"?

- This laptop is a *great deal*.
- A *great deal* of media attention surrounded the release of the new laptop.
- This laptop is a *great deal* ... and I've got a nice bridge you might be interested in.
- This film should be brilliant. It sounds like a great plot, the actors are first grade, and the supporting cast is good as well, and Stallone is attempting to deliver a good performance. However, it can't hold up.

Pawan Goyal (IIT Kharagpur)      Sentiment Analysis - Introduction      Week 12, Lecture 1

So, for that you have to see how people write about in their comments. So, this is one example, this laptop is a great deal. If you see the sentence you will say yes this looks like a positive review. But now let us see it in different like, a great deal of media attention surrounded the release of the new laptop. I am using the same words right great and deal, and this sentence also contains great and deal also contains the laptop. Would you say this is a positive sentiment? the sentence looks like neutral. So, you say ok. There is a lot of media attention, but says nothing about the laptop that whether the laptop is good or bad while the previous laptop was doing that. So, a case does not a stop here.

People will also write like that this laptop is a great deal and I have got a nice bridge you might be interested in right. So, this is some sort of phrase people use to say to use sarcasm, this is if this nice deal nice deal (Refer Time: 25:11) that you might be interested in so that means so this is like a really bad laptop. If you are talking is nice still, this is like a it is like a joke. So, then the text can also contains some humor or sarcasm. So, how do you find that this is not a positive sentiment, although there is no negative word in this sentence?

Similarly, here the film should be brilliant, it sounds like a great plot, that the actors are first grade; supporting cast is good as well. And Stallone is attempting to deliver a good performance everything positive. However, it cannot hold up; and the last sentence immediately turns all the polarity to negative. So, it says although it contains lot of positive words because of this is all negative. So, this that is what makes sentiment analysis much more difficult that you cannot do it easily by using simple features.

So, this was about the introduction. So, now, what we will do we will talk about different method that can be used. Again the most in common method is using some sort of known words that are having positive or negative polarity. And you have some sort of large list of these words not manually created some 10 or 12 words, it can be some 1000 of words that people have annotated with positive or negative sentiments, and this is called the affective lexicons. So, you can use these lexicons and they are readily available. So, we will see those. And you can use that for your task.

And then we will go further into how do you learn these lexicons and so on. So that is for this lecture, and then we will talk about this in the next lecture.

Thank you.

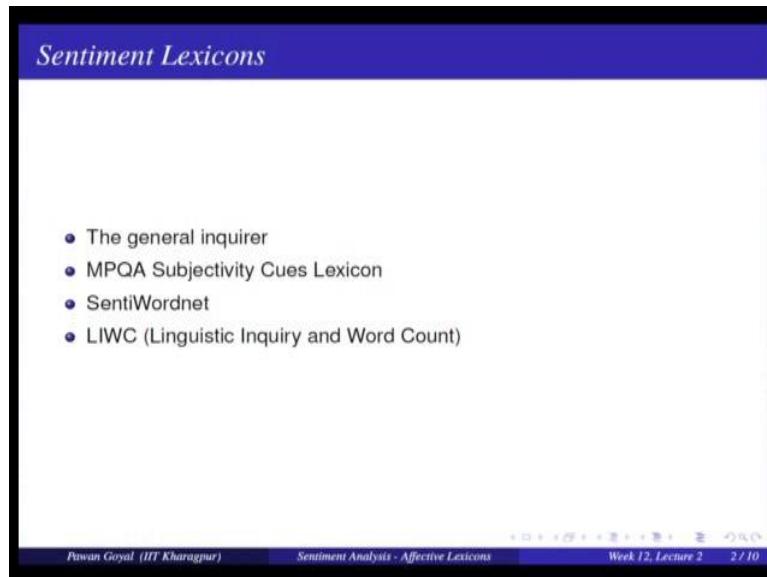
**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 62**  
**Sentiment Analysis - Affective Lexicons**

Welcome back for the second lecture of this week. So, we talked about Sentiment Analysis. And we were saying that the one of the most important tools for doing sentiment analysis is using some sort of Lexicons. So, were some people have manually annotated some words whether they are positive or negative and so on, and given a new text you can use that to find out what is the major polarity and you can use some complicated models also to see how it changes by using communications and so on.

But important is finding out whether the some of the words are having positive or negative polarity. And that can be obtained by using sentiment lexicons. So, there are many different lexicons.

(Refer Slide Time: 00:59)



In this lecture I will just give some introduction to some of these and they are all available and you can go and download those in using your applications. So for example, the general inquirer is one of the most more popular lexicons, then MPQ; multi prospective question answering subjectivity cues lexicon, SentiWordnet is something that has been used widely and then in the recent times the tool LIWC that is linguistic inquiry and word count has been

popularly used for doing sentiment analysis along with some many other task. So, it has been very very popular.

So, let us have a look at these tools. Interestingly the tools I will be showing are mostly built for English there are efforts and on using these tools took good strap lexicons in other languages and many such efforts have actually been able to build such lexicons and such tools. But there is a lot of scope on for doing that in say Indian languages. Some of the tools available and many of these are not available. And if you have to process comments etcetera in Indian context you might want to use to build some of these also. So, we will also see how do you learn these in the next lecture.

(Refer Slide Time: 02:21)

The General Inquirer																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
Categories																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
<ul style="list-style-type: none"> <li>Positive (1915 words) and Negative (2291) words</li> <li>Strong vs weak, active vs passive, overstated vs understated</li> <li>pleasure, pain, virtue, vice, motivation, cognitive orientation etc.</li> </ul>																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
<table border="1"> <thead> <tr> <th>Entry</th><th>Source</th><th>Positiv</th><th>Negativ</th><th>Pstv</th><th>Affl</th><th>Ngtv</th><th>Hostile</th><th>Strong</th><th>Power</th><th>Weak</th><th>Submit</th><th>Active</th></tr> </thead> <tbody> <tr> <td>ABANDON</td><td>H&amp;L.vd</td><td></td><td>Negative</td><td></td><td></td><td>Ngtv</td><td></td><td></td><td></td><td>Weak</td><td></td><td></td></tr> <tr> <td>ABANDONMENT</td><td>H&amp;L</td><td></td><td>Negative</td><td></td><td></td><td></td><td></td><td></td><td></td><td>Weak</td><td></td><td></td></tr> <tr> <td>ABATE</td><td>H&amp;L.vd</td><td></td><td>Negative</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABDUCE</td><td>H&amp;L</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABDUCT</td><td>H&amp;L</td><td></td><td>Negative</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABDICTATE</td><td>H&amp;L</td><td></td><td>Negative</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABDICTED</td><td>H&amp;L</td><td></td><td>Negative</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABHOR</td><td>H&amp;L</td><td></td><td>Negative</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABIDE</td><td>H&amp;L</td><td>Positive</td><td>Positive</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABJECT</td><td>H&amp;L.vd</td><td></td><td>Negative</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABLE</td><td>H&amp;L.vd</td><td>Positive</td><td></td><td>Pstv</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABNORMAL</td><td>H&amp;L.vd</td><td></td><td>Negative</td><td></td><td></td><td>Ngtv</td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABORD</td><td>H&amp;L.vd</td><td></td><td>Negative</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABROBISH</td><td>H&amp;L.vd</td><td></td><td>Negative</td><td></td><td></td><td>Ngtv</td><td>Hostile</td><td>Strong</td><td>Power</td><td></td><td></td><td></td></tr> <tr> <td>ABROBITION</td><td>H&amp;L</td><td></td><td>Negative</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABROBITIONABLE</td><td>H&amp;L</td><td></td><td>Negative</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABROBITIONER</td><td>H&amp;L</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABROBUND</td><td>H&amp;L</td><td>Positive</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#1</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#2</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#3</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#4</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#5</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#6</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#7</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#8</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#9</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#10</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#11</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#12</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#13</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#14</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#15</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#16</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#17</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#18</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#19</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#20</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#21</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#22</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#23</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#24</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#25</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#26</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#27</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#28</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#29</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#30</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#31</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#32</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#33</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#34</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#35</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#36</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#37</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#38</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#39</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#40</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#41</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#42</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#43</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#44</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#45</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#46</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#47</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#48</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#49</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#50</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#51</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#52</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#53</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#54</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#55</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#56</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#57</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#58</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#59</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#60</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#61</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#62</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#63</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#64</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#65</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#66</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#67</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#68</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#69</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#70</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#71</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#72</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#73</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#74</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#75</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#76</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#77</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#78</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#79</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#80</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#81</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#82</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#83</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#84</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#85</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#86</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#87</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#88</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#89</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#90</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#91</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#92</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#93</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#94</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#95</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#96</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#97</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#98</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#99</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#100</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#101</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#102</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#103</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#104</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#105</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#106</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#107</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#108</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#109</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#110</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#111</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#112</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#113</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#114</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#115</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#116</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#117</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#118</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#119</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#120</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#121</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#122</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#123</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#124</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#125</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#126</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#127</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#128</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#129</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#130</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#131</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#132</td><td>H&amp;L.vd</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td>ABOUT#133</td><td>H&amp;L.vd</td><td></td><td></td><td></td>&lt;td</tr></tbody></table>	Entry	Source	Positiv	Negativ	Pstv	Affl	Ngtv	Hostile	Strong	Power	Weak	Submit	Active	ABANDON	H&L.vd		Negative			Ngtv				Weak			ABANDONMENT	H&L		Negative							Weak			ABATE	H&L.vd		Negative										ABDUCE	H&L												ABDUCT	H&L		Negative										ABDICTATE	H&L		Negative										ABDICTED	H&L		Negative										ABHOR	H&L		Negative										ABIDE	H&L	Positive	Positive										ABJECT	H&L.vd		Negative										ABLE	H&L.vd	Positive		Pstv									ABNORMAL	H&L.vd		Negative			Ngtv							ABORD	H&L.vd		Negative										ABROBISH	H&L.vd		Negative			Ngtv	Hostile	Strong	Power				ABROBITION	H&L		Negative										ABROBITIONABLE	H&L		Negative										ABROBITIONER	H&L												ABROBUND	H&L	Positive											ABOUT#1	H&L.vd												ABOUT#2	H&L.vd												ABOUT#3	H&L.vd												ABOUT#4	H&L.vd												ABOUT#5	H&L.vd												ABOUT#6	H&L.vd												ABOUT#7	H&L.vd												ABOUT#8	H&L.vd												ABOUT#9	H&L.vd												ABOUT#10	H&L.vd												ABOUT#11	H&L.vd												ABOUT#12	H&L.vd												ABOUT#13	H&L.vd												ABOUT#14	H&L.vd												ABOUT#15	H&L.vd												ABOUT#16	H&L.vd												ABOUT#17	H&L.vd												ABOUT#18	H&L.vd												ABOUT#19	H&L.vd												ABOUT#20	H&L.vd												ABOUT#21	H&L.vd												ABOUT#22	H&L.vd												ABOUT#23	H&L.vd												ABOUT#24	H&L.vd												ABOUT#25	H&L.vd												ABOUT#26	H&L.vd												ABOUT#27	H&L.vd												ABOUT#28	H&L.vd												ABOUT#29	H&L.vd												ABOUT#30	H&L.vd												ABOUT#31	H&L.vd												ABOUT#32	H&L.vd												ABOUT#33	H&L.vd												ABOUT#34	H&L.vd												ABOUT#35	H&L.vd												ABOUT#36	H&L.vd												ABOUT#37	H&L.vd												ABOUT#38	H&L.vd												ABOUT#39	H&L.vd												ABOUT#40	H&L.vd												ABOUT#41	H&L.vd												ABOUT#42	H&L.vd												ABOUT#43	H&L.vd												ABOUT#44	H&L.vd												ABOUT#45	H&L.vd												ABOUT#46	H&L.vd												ABOUT#47	H&L.vd												ABOUT#48	H&L.vd												ABOUT#49	H&L.vd												ABOUT#50	H&L.vd												ABOUT#51	H&L.vd												ABOUT#52	H&L.vd												ABOUT#53	H&L.vd												ABOUT#54	H&L.vd												ABOUT#55	H&L.vd												ABOUT#56	H&L.vd												ABOUT#57	H&L.vd												ABOUT#58	H&L.vd												ABOUT#59	H&L.vd												ABOUT#60	H&L.vd												ABOUT#61	H&L.vd												ABOUT#62	H&L.vd												ABOUT#63	H&L.vd												ABOUT#64	H&L.vd												ABOUT#65	H&L.vd												ABOUT#66	H&L.vd												ABOUT#67	H&L.vd												ABOUT#68	H&L.vd												ABOUT#69	H&L.vd												ABOUT#70	H&L.vd												ABOUT#71	H&L.vd												ABOUT#72	H&L.vd												ABOUT#73	H&L.vd												ABOUT#74	H&L.vd												ABOUT#75	H&L.vd												ABOUT#76	H&L.vd												ABOUT#77	H&L.vd												ABOUT#78	H&L.vd												ABOUT#79	H&L.vd												ABOUT#80	H&L.vd												ABOUT#81	H&L.vd												ABOUT#82	H&L.vd												ABOUT#83	H&L.vd												ABOUT#84	H&L.vd												ABOUT#85	H&L.vd												ABOUT#86	H&L.vd												ABOUT#87	H&L.vd												ABOUT#88	H&L.vd												ABOUT#89	H&L.vd												ABOUT#90	H&L.vd												ABOUT#91	H&L.vd												ABOUT#92	H&L.vd												ABOUT#93	H&L.vd												ABOUT#94	H&L.vd												ABOUT#95	H&L.vd												ABOUT#96	H&L.vd												ABOUT#97	H&L.vd												ABOUT#98	H&L.vd												ABOUT#99	H&L.vd												ABOUT#100	H&L.vd												ABOUT#101	H&L.vd												ABOUT#102	H&L.vd												ABOUT#103	H&L.vd												ABOUT#104	H&L.vd												ABOUT#105	H&L.vd												ABOUT#106	H&L.vd												ABOUT#107	H&L.vd												ABOUT#108	H&L.vd												ABOUT#109	H&L.vd												ABOUT#110	H&L.vd												ABOUT#111	H&L.vd												ABOUT#112	H&L.vd												ABOUT#113	H&L.vd												ABOUT#114	H&L.vd												ABOUT#115	H&L.vd												ABOUT#116	H&L.vd												ABOUT#117	H&L.vd												ABOUT#118	H&L.vd												ABOUT#119	H&L.vd												ABOUT#120	H&L.vd												ABOUT#121	H&L.vd												ABOUT#122	H&L.vd												ABOUT#123	H&L.vd												ABOUT#124	H&L.vd												ABOUT#125	H&L.vd												ABOUT#126	H&L.vd												ABOUT#127	H&L.vd												ABOUT#128	H&L.vd												ABOUT#129	H&L.vd												ABOUT#130	H&L.vd												ABOUT#131	H&L.vd												ABOUT#132	H&L.vd												ABOUT#133	H&L.vd			
Entry	Source	Positiv	Negativ	Pstv	Affl	Ngtv	Hostile	Strong	Power	Weak	Submit	Active																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
ABANDON	H&L.vd		Negative			Ngtv				Weak																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
ABANDONMENT	H&L		Negative							Weak																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
ABATE	H&L.vd		Negative																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
ABDUCE	H&L																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABDUCT	H&L		Negative																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
ABDICTATE	H&L		Negative																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
ABDICTED	H&L		Negative																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
ABHOR	H&L		Negative																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
ABIDE	H&L	Positive	Positive																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
ABJECT	H&L.vd		Negative																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
ABLE	H&L.vd	Positive		Pstv																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
ABNORMAL	H&L.vd		Negative			Ngtv																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
ABORD	H&L.vd		Negative																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
ABROBISH	H&L.vd		Negative			Ngtv	Hostile	Strong	Power																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
ABROBITION	H&L		Negative																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
ABROBITIONABLE	H&L		Negative																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
ABROBITIONER	H&L																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABROBUND	H&L	Positive																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
ABOUT#1	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#2	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#3	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#4	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#5	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#6	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#7	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#8	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#9	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#10	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#11	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#12	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#13	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#14	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#15	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#16	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#17	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#18	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#19	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#20	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#21	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#22	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#23	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#24	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#25	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#26	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#27	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#28	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#29	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#30	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#31	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#32	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#33	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#34	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#35	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#36	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#37	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#38	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#39	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#40	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#41	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#42	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#43	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#44	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#45	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#46	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#47	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#48	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#49	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#50	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#51	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#52	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#53	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#54	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#55	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#56	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#57	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#58	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#59	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#60	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#61	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#62	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#63	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#64	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#65	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#66	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#67	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#68	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#69	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#70	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#71	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#72	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#73	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#74	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#75	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#76	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#77	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#78	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#79	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#80	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#81	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#82	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#83	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#84	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#85	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#86	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#87	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#88	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#89	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#90	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#91	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#92	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#93	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#94	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#95	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#96	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#97	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#98	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#99	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#100	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#101	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#102	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#103	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#104	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#105	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#106	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#107	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#108	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#109	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#110	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#111	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#112	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#113	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#114	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#115	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#116	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#117	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#118	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#119	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#120	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#121	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#122	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#123	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#124	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#125	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#126	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#127	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#128	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#129	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#130	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#131	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#132	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
ABOUT#133	H&L.vd																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															

are all negative words. And then you get some positive words like ability, able, abound, abide; these are all positive words. Then there are other dimensions like weak and strong. Is it a weak word or strong word? So, let us see weak versus strong. So, ability is a strong indicator, able strong, abolish is strong; but abdicate, abide, amendment are not very strong so they are weak.

So, this is also annotated this is weak sentiment this is an strong sentiment. And then they are other dimensions if you can see the other columns of this spreadsheet. And this to this comes readily available and you can use these for any you can see how to use that in your different tasks.

Then I was saying SentiWordnet is again very very popular. So, what was this effect? You know we talked about ordinary in this course. So, WordNet has lot of different synsets. So, what was done in SentiWordnet each of this synsets were taken and people, so by some way of finally doing manual supervisor supervision they try to label what is the sentiment score of this particular synset. Is it positive, negative? What is the score? What is the objectivity, subjectivity? So, they try to manually label that.

(Refer Slide Time: 04:45)

The screenshot shows the SentiWordNet homepage. At the top, it says "SentiWordNet". Below that, there's a purple sidebar with the following text:

- Home page: <http://sentiwordnet.isti.cnr.it/>
- All Wordnet synsets automatically annotated for degrees of positivity, negativity, and neutrality/objectiveness

Below the sidebar, there's a light blue box labeled "Example". It contains two entries:

- estimable (J.3)* : "may be computed or estimated"  
Pos 0 Neg 0 Obj 1
- estimable (J.1)* : "deserving of respect or high regard"  
Pos .75 Neg 0 Obj .25

At the bottom of the slide, there's a footer with the names "Pawan Goyal (IIT Kharagpur)", "Sentiment Analysis - Affective Lexicons", and "Week 12, Lec 1". To the right of the footer, there's a circular profile picture of a man.

So, they took all WordNet synsets and they were automatically annotated for degrees of positive, negativity and neutrality and objectiveness; so examples are like that. So, you have two different synsets of the word estimable, so 3 and 1. So, J 3 is may be computed or estimated you can see that this is like a more objective does not have any sentiment, so it has

positive 0 negative 0 objective 1.

On the other hand the second sense deserving of respect or high regard it looks like positive. So, positive is 0.75 negative is 0 and objectivity is 0.25. They have all these three scores.

(Refer Slide Time: 05:28)

The screenshot shows a presentation slide with a blue header bar containing the title 'Other Lexicons'. Below the header, there are two main sections: 'MPQA Subjectivity Cues Lexicon' and 'Bing Liu Opinion Lexicon'.  
**MPQA Subjectivity Cues Lexicon**

- Home page: [http://www.cs.pitt.edu/mpqa/subj\\_lexicon.html](http://www.cs.pitt.edu/mpqa/subj_lexicon.html)
- 6885 words from 8221 lemmas: 2718 positive, 4912 negative
- Each word annotated for intensity (strong, weak)

  
**Bing Liu Opinion Lexicon**

- Bing Liu's Page on Opinion Mining
- <http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar>
- 6786 word: 2006 positive, 4780 negat...<http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar>

At the bottom of the slide, there is footer information: 'Pawan Goyal (IIT Kharagpur)', 'Sentiment Analysis - Affective Lexicons', 'Week 12, Lecture 2', and '5 / 10'.

Then another important lexicon is this MPQA subjectivity lexicon. Again you can go to this website where you can download this lexicon. So, it contains around 6885 words from 8000 plus lemmas and 2700 are positive and 4900 are negative. And each word has been labeled also for its intensity. So, in what are positive words, what are negative word, but you also know the intensity; is it strong is it weak and so on.

Then there is opinion lexicon by Bing Liu- that you can get from this the page on opinion by Bing Liu and there are again around refer the same words and some positive and some negative words. So, you can go and download these data sets and see whether this can be helpful for certain tasks.

(Refer Slide Time: 06:31)

The screenshot shows a presentation slide titled "LIWC (Linguistic Inquiry and Word Count)". The content includes:

- Home page: <http://www.liwc.net/>
- 2300 words, > 70 classes

**Affective Processes**

- Negative emotion (bad, weird, hate, problem, tough)
- Positive emotion (love, nice, sweet)

**Cognitive Processes**

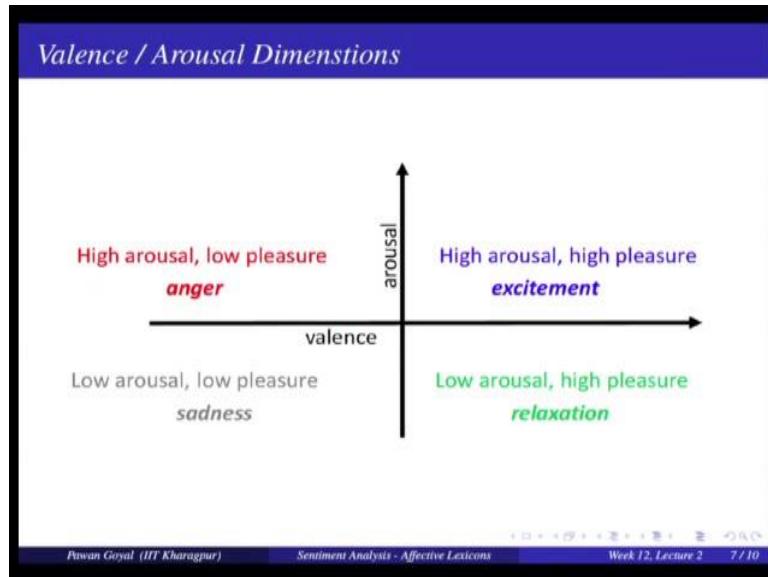
- Tentative (maybe, perhaps, guess), Inhibition (block, constraint)
- Comes with a small fee

At the bottom, there are navigation icons and text: "Ptwan Goyal (IIT Kharagpur)", "Sentiment Analysis - Affective Lexicons", "Week 12, Lecture 2", and "6 / 10".

And then I was saying in recent times another tool that has become very popular is the LIWC tool; linguistic enquiry word count. So, what is interesting is that. Again you can go to this webpage and explore what does this tool does. So, it takes 2300 different words and divides them into 70 different classes. And these classes range from simple positive negative to many different social things also. Like, so you are having negative emotions. So, you will say words like bad, weird, hate, problem, tough; they will be getting negative emotions. And love, nice, sweet are getting positive emotions.

But you will also see some cognitive processes like; some words denote some tentativeness like here, maybe, perhaps, guess; they might indicate in addition like block, constraint and so on. So, if you will see there are 70 different classes in which words are explored. And these classes make it much more richer representation that can be used for task like sentiment analysis also seeing other sort of effective states that we talked about earlier in the last week. So, we can talk about emotions, we can talk about inter personal things and so on by using this tool. Only thing is that it comes with a small fee, say if you have to use that you have to pay some small fee.

(Refer Slide Time: 07:51)



So, in general when we talk about sentiment you can also talk about two different aspects: one is called the valence another is called the arousal. So, by valence I mean positive or negative; how positive it is, how negative it is. And arousal is kind of strength. So, arousal means yes this is very very strong sentiment and low arousal means say it is not it is a weak sentiment.

So, you can put them like a like an access here, so you have some quadrant this is like valence going from negative to positive and arousal going from low to high and you can see ok. So, here words are high arousal, high pleasure like excitement it has been having valence is positive for excitement and the arousal is also high. And you can accordingly think of a word that is positive, but arousal is low like getting relaxed, so relaxation. This is having again positive, but arousal is low.

Similarly negative you can take the contrast between anger and sadness. Anger and sadness both are having the negative valence, but in terms of arousals anger is in on high and sadness is low. There are again attempts that have put words in these two dimensions that what is the valence and what is the arousal of these words.

(Refer Slide Time: 09:14)

*Lexicon of valence, arousal, and dominance*

- Warriner, Amy Beth, Victor Kuperman, and Marc Brysbaert. "Norms of valence, arousal, and dominance for 13,915 English lemmas." *Behavior research methods* 45.4 (2013): 1191-1207.
- **Supplementary data:** This word is licenced under a Creative Commons.

*Ratings for 14,000 words for emotional dimensions:*

- **valence** (the pleasantness of the stimulus)
- **arousal** (the intensity of emotion provoked by the stimulus)
- **dominance** (the degree of control exerted by the stimulus)

Pawan Goyal (IIT Kharagpur)      Sentiment Analysis - Affective Lexicons      Week 12, Lecture 2      9 / 10

So, this is the work; so lexicon of valence arousal and dominance; so norms of valence arousal dominance for 13915 English lemmas. And this word is licensed under a creative commons license it is available for non commercial use.

So, what they have done? They have taken roughly 14000 words and they have labeled them for the valence that is how pleasant or unpleasant this is arousal. So, what is the intensity of the emotion? So whether it is highly arousal or low; and what is the dominance? So, what is the degree of control by the stimulus? So, these 14000 words are labeled with these three.

(Refer Slide Time: 09:59)

*Lexicon of valence, arousal, and dominance*

*valence (the pleasantness of the stimulus)*

- 9: happy, pleased, satisfied, contented, hopeful
- 1: unhappy, annoyed, unsatisfied, melancholic, despaired, or bored

*arousal (the intensity of emotion provoked by the stimulus)*

- 9: stimulated, excited, frenzied, jittery, wide-awake, or aroused
- 1: relaxed, calm, sluggish, dull, sleepy, or unaroused

*dominance (the degree of control exerted by the stimulus)*

- 9: in control, influential, important, dominant, autonomous, or controlling
- 1: controlled, influenced, cared-for, awed, submissive, or guided

Pawan Goyal (IIT Kharagpur)      Sentiment Analysis - Affective Lexicons      Week 12, Lecture 2      9 / 10

Let us see some example of the valence. So, how pleasant the stimulus is? So these are the words with valence of 9 and with the 1. And this you can easily see these are like what we are simply talking sentiment analysis. So, 9 words are; happy, pleased, satisfied, contended, hopeful, they are all 9. And 1 here; unhappy, annoyed, unsatisfied, despaired and bored and so on; so these are all in the valence.

Then if you go to arousal; again 9 they are like stimulated, excited, frenzied, and aroused; and 1 you will have words like relaxed, calm, sluggish, dull, and sleepy. So, you can see the degree of arousal also. Then there are; and you should not dominant. So, like in control, influential, important, dominant they are all having a reading of 9; and on 1 you will have controlled, influenced like controlled by someone else, cared-for, awed, submissive, (Refer Time: 11:02) low control.

(Refer Slide Time: 11:05)

Lexicon of valence, arousal, and dominance: Examples					
Valence		Arousal		Dominance	
vacation	8.53	rampage	7.56	self	7.74
happy	8.47	tornado	7.45	incredible	7.74
whistle	5.7	zucchini	4.18	skillet	5.33
conscious	5.53	dressy	4.15	concur	5.29
torture	1.4	dull	1.67	earthquake	2.14

Pawan Goyal (IIT Kharagpur)
Sentiment Analysis - Affective Lexicons
Week 12, Lecture 2
10 / 10

So, here are some other examples with the words and the actual fraction with this they are labeled. So, vacation is getting a high valence. Yes happy gets a high valence, whistle gets (Refer Time: 11:18), conscious 5, torture gets 1.4. Rampage gets 7.5, arousal high tornado gets high, zucchini gets around 4, dressy and dull gets low. And similarly here self incredible get very high dominance; on the other hand words like earthquake get a very low dominance. So, these words are all labeled with all these dimensions.

With that there are many other sort of lexicons also that people have used and they also around on the web. There are some of the more important ones and most of these are

available for download and you can use in your different tasks. But many of these lexicons were not created by manual labeling. Some of these were created automatically and someone then manually verified.

So, what we will see in the next lecture that is there is some way in which you can create these lexicons on your own by using some of the techniques and we talked about; by doing some sort of good strapping or by using some sort of (Refer Time: 12:23) can be learn these lexicons from the data. And this is again very interesting idea and many lexicons have been (Refer Time: 12:33) by using that method, so that you will talk about in the next lecture.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 63**  
**Learning Affective Lexicons**

Hello everyone. Welcome to the third lecture of this week. So, we were talking about sentiment analysis. And in the last lecture, we talked about various sentiment lexicons that are available to us and that you can use for various task related to sentiment analysis. So, in this lecture, what we will be doing we will be seen suppose you have to learn these lexicons on your own without having to do manually labeling each word with the sentiment polarity, so what can be some possible approaches you can take. So, while you will see that most of the lexicons are built for English language they not too many works for other languages.

So, suppose you want to build a lexicon for your own language, it can be Indian language, for example, so you can want to build for Hindi, Bengali and or Tamil, so how do you start approaching this problem. So, you can always take all the words and have people manually annotate that, but is there some automated approach and that is why you will also see some of the concepts that we have talked in this course how they can be useful in doing this task.

(Refer Slide Time: 01:19)

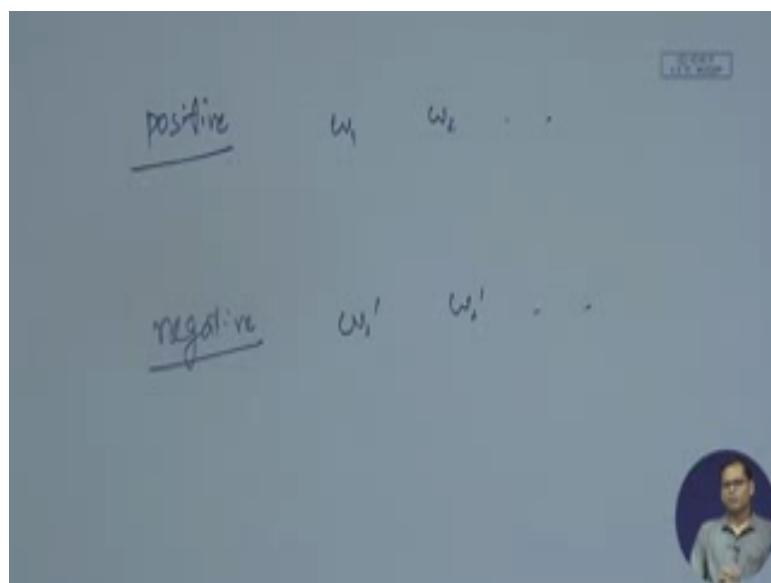
So, when suppose I have to built, I have to learn these the polarity of different words in my lexicon. So, what might be the basic idea intuition that I can start with. And then intuition can

come again from some sort of distribution hypothesis in that when to the words of say simple idea get to together or how do the words of different polarity are get together. So, for example, here we are saying think of the adjectives that are joined by and. So, will they have the same polarity?

Suppose, you are having a sentence and you see fair and legitimate. So, you can say probably both these words have same polarity. You may not be able to tell; what is the polarity of the individual word, but if they occurring with and fair and legitimate, you would be able to see that they will have the same polarity. Same here, corrupt and brutal these two words have the same polarity. Now, can you have some similar indication for saying whether the two words will have different polarity? So, think about cases where you say, but and so on like here it is fair, but brutal. So, you will know that the polarity of both the words fair and brutal will be opposite to each other.

Now can this sort of simple intuition can help us in learning the sentiment lexicons. So, what will you have to do if you have to use some methods like that? So, we can see that we cannot find out probably what is the polarity of an individual word by this method, but if I know the polarity of a single word, I might be able to infer the same for other words if they occurring with some sort of connectives like and, or, but. So, what should I start with?

(Refer Slide Time: 03:04)



So, what I might be doing I might be starting with list of positive words and some list of negative words positive and negative sentiment words. So, I take some w 1, w 2 so on w 1

prime w 2 prime and so on. So, this might be some very simple list that you start with that you might have created manually. So, what you will be doing next, you know there the words that I have in the same polarity might occur in a corpus with certain connectives.

So, now you will go to a large corpus, it can be web or any of the corpus that you have and try to find out these if these words are occurring with other words with some connectives. And for that you can just search like fair and, you can search for this and wherever you find fair and x you might assign a positive polarity to x. Similarly, fair, but and you find fair, but brutal and you might assign a negative polarity to that so that can even possible approach.

(Refer Slide Time: 04:00)

The screenshot shows a presentation slide with a blue header bar containing the title 'Learning Sentiment Lexicons'. Below the header is a purple box containing the text 'Step 1: Label seed set of adjectives'. Inside this box, there are two bullet points: one for 'Positive cases' (adequate, central, clever, famous, intelligent, remarkable, reputed, sensitive, slender, thriving ...) and one for 'Negative cases' (contagious, drunken, ignorant, lanky, listless, primitive, strident, troublesome, unresolved, unsuspecting ...). At the bottom of the slide, there is a video player interface with a circular video thumbnail showing a man's face, and a progress bar indicating the video is at 0:00.

So, I start with some seed set of adjectives that I can label manually or I can get it from some sort of it can be obtained also from some websites give me some positive and negative adjectives. Now, so suppose I got some positive cases adequate, central, clever, famous; and negative cases like contagious, drunken, ignorant, and so on. So, now I go to my corpus and try to search these with some connectives added to these.

(Refer Slide Time: 04:30)

The screenshot shows a search results page from a web browser. The search query is "was adequate and". The results are filtered by "Web" and show approximately 16,100,000 results. One result is highlighted, showing a review from TripAdvisor: "The room was adequate and clean. The pool area was very ...". Below the search bar, there are tabs for "Web", "Image", "Videos", and "More". At the bottom of the slide, there is a navigation bar with "Powerpoint (MIT Kharragger)", "Learning Affective Lexicons", and "Week 2.2: Lexicons".

So, like here, so I want to expand my seed set to conjoined adjectives. So, I can search say on web. So, I will say was adequate and, so that way I will find out the words that are having a possible ideas similar to adequate. So, suppose I search that I find some results like this. The room was adequate and clean and immediately, I will say clean will probably have a positive polarity tool.

(Refer Slide Time: 04:57)

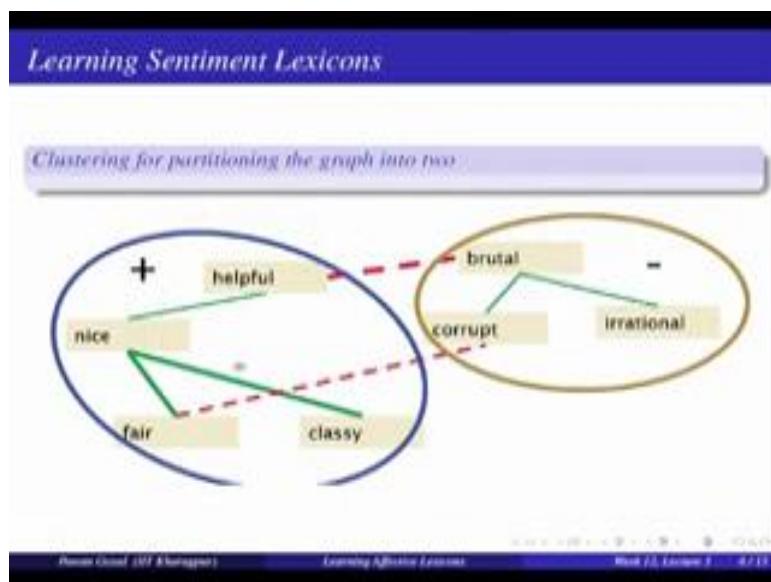
The screenshot shows a graph visualization titled "Step 3: Construct a graph". It displays a network of words represented as nodes: nice, helpful, brutal, corrupt, irrational, fair, and classy. The nodes are colored in shades of yellow and orange. Edges connect the nodes, with some edges being solid green lines and others dashed red lines, representing different types of polarity similarity or relationships. The graph is centered on the word "nice". Below the graph, there is a text box stating "Polarity similarity is assigned to each word pair;". At the bottom of the slide, there is a navigation bar with "Powerpoint (MIT Kharragger)", "Learning Affective Lexicons", and "Week 2.2: Lexicons".

And if I do that if I apply this method over all my seed sets, also from the seed set I get other adjectives, you can continue applying this method and you can obtain a graph like that. So,

what I using in this graph you find that nice and fair occur with some connectives, so that is why there is an edge green edge and they are up appearing with the connective like and that means, same polarity. Similarly, here nice and helpful occurring with connective end, nice and classy occurring with connective end, fair classy do not occur together, but that may not be required.

Similarly, you find brutal and corrupt, they occur when this with the connective, end brutal and irrational occur with the connective end, but there are some red dashed lines also, what are these so that would be suppose towards are occurring with the connective of, but so helpful, but brutal. So, then you say they have opposed polarity. So, now, that is how you will label the edges in your graph. So, some are edges that see ok they have similar pole polarity some edges that is why they have polarity and now once you have built this graph over a large set of words you can apply some clustering algorithm. To say which of these words are having green edges with among each other, but no red edges that can be some criteria by which you can cluster these words. And this can finally, give you positive and negative examples from the data.

(Refer Slide Time: 06:30)



So, that is the partition of the two graphs and different graphs give you positive as well as negative sentiment lexicons that can be the standard method that to be applied over any language once you have some seed set of sentiment words and you have a corpus at your hand. And you can always increase your set of rules that you are having so right now you are

using only and or, but you find out some other rules by which different sentiment words can be connected, and you apply those rules also to get different edges in your graph. So, once you apply this method from a real experiment what kind of lexicon did they obtain.

(Refer Slide Time: 07:13)

The screenshot shows a presentation slide with a blue header bar containing the title 'Output Polarity Lexicon'. Below the header, there are two main sections: 'Positive' and 'Negative'.  
**Positive:** bold decisive disturbing generous good honest important large mature patient peaceful positive proud sound stimulating straightforward strange talented vigorous witty ...  
**Negative:** ambiguous cautious cynical evasive harmful hypocritical inefficient insecure irrational irresponsible minor outspoken pleasant reckless risky selfish tedious unsupported vulnerable wasteful ...

You see the positive words the obtained words like bold this is disturbing, generous, good, honest, important, large, mature and so on. These are the words that they obtain obtained this will be noisy this may not be very, very good set, but the ideas is that you will get a lot of good examples right. So, is like generous, good, honest, important, large, mature, peaceful, they look all having they all look like having a positive sentiment. The only exception looks like disturbing here they may be some other little.

Similarly, that these are the words that they obtain for negative sentiment ambiguous, cautions, cynical, evasive, harmful, hypocritical, and all these look the words with negative sentiment. So, this was obtained by giving a seed set, seed label to only to a few words, not these words. And these when you found out the words with different connectives, you are able to obtain this positive and negative class. And there will be some erroneous cases like disturbing, coming as positive example, cautions; outspoken pleasant coming as negative example that few cases might be there, but overall this get good precision.

And if you have a very good corpus, good seed set you might also get a very good (Refer Time: 08:29) by this method. So, that is one simple method that you can apply to some sort of bootstraps your sentimental lexicon.

(Refer Slide Time: 08:39)

The slide has a blue header bar with the title "Turney Algorithm". Below the header is a large white area containing a bulleted list of three items:

- Extract a *phrasal lexicon* from reviews
- Learn polarity of each phrase
- Rate a review by the average polarity of its phrases

At the bottom of the slide, there is a small video player interface. It shows a circular thumbnail of a man, with the text "Powerpoint (MP4, 1000x600)" to the left and "Learning Affection Features" to the right. On the far right, it says "Week 2.2, Lecture 2".

Now so there are some other algorithms also propose a literature for doing this task. So, we will see some of these. So, let us look at the Turney algorithm. So, what did this algorithm? So, what this algorithm did. So, they took some example from reviews. So, they start with the data first, they did not start with seed set of labeled words with emotions opinions, they start with the data set. And they said because you got a lot of opinions from review data set. So, let us take some review data sets. Now, in the review data set they are so what the hypothesis was so opinions will be expressed by some sort of noun phrases, there are some adjectives they are some nouns and there will be opinions associate with those.

So, let us start with extracting these noun phrases that are occurring a lot in this corpus. Once we have a good set of noun phrases then I will try to find out some sort of sentiment associate with those. So, let us see how they did that. So, you first extract a phrasal lexicon from reviews, and then try to learn polarity for each phrase here. Once you have learned the polarity, you might go to a next task that is find out what will be the rating of this review. Suppose, this is a plane text there is no rating, you can use the sentiment from faces in the review to give the rating to the review, whether it is a positive review or negative review.

(Refer Slide Time: 10:06)

Extract two-word phrases with adjectives		
First Word	Second Word	Third Word (not extracted)
JJ	NN or NNS	anything
RB, RBR, RBS	JJ	Not NN nor NNS
JJ	JJ	Not NN or NNS
NN or NNS	JJ	Not NN nor NNS
RB, RBR, or RBS	VB, VBD, VBN, VBG	anything

So, interesting thing was how to start building their phrasal lexicon. So, it did not they did not take any noun phrase that occurs in the review. So, they made some patterns by manually seeing how the important sentiment opinion phrases occur in text. So, what kinds of patters are there? And they built some patterns like the first word should be a JJ an adjective second word can be a NN or NNS. And third word not extracted is anything that is whenever they find first word as an adjective part of speech text second word with NN or NNS, they will extract if there free the lexicon irrespective of what is the third word they will do with all the corpus. Like they built other rules like if it is in adverb, second word is an adjective, and third word is not noun or NNS, they will take it in their phrasal lexicon.

Similarly, both first and second word are adjectives third word is not NN or NNS, they will take it. First word is NN or NNS, second word is adjective and then the next word is not NN or NNS again they will take it in their phrasal lexicon and like that they had another rule adjective and a verb. So, whatever phrases in my lexicon or the conjugative words my lexicon were following these regular expressions defined to a part of speech text they took that in their phrasal lexicon. Again that is a very interesting method you can take some examples on your own and find out what is the usual pattern of part of speech sector that occurs and according you define that is how you will extract your phrasal lexicon.

So, now once they extract their phrasal lexicon, the next task would be how do they give them some sentiment score. So, what the hypothesis they were saying some of these will

positive some of these be negative. So, whichever are positive will probably occur with a word like excellent; and whichever negative will occur with words like poor. So, now how do we measure this co occurrence? So, how many times it is occurring with excellent how many times it is occurring with poor and use, that to give a sentiment polarity to each of these word in the lexicon.

So, again here you can use the corpus. So, you can take a large corpus find out how many times each of these phrases occurring with excellence and poor, but simple count will not work. So, we have to use a very sophisticated method. So, do you remember any such measure that we discussed in this course. So, to find out how common do they do towards co occur. So, if you remember we talked about point wise mutual information, so you can find out the point wise mutual information of each word with these excellent and poor.

(Refer Slide Time: 13:02)

*Measuring the polarity of the phrases*

- Positive phrases co-occur more with "excellent"
- Negative phrases co-occur more with "poor"
- How to measure the co-occurrence?

*Pointwise Mutual Information*

$$PMI(X, Y) = \log_2 \frac{P(x, y)}{P(x)P(y)}$$

$$PMI(X, Y) = \log_2 (P(x, y)) / (P(x)P(y))$$

So, I will use this method to find out what is the PMI between a phrasal lexicon and excellent phrasal word and excellent and phrasal word and poor. So, how did they find this PMI value, you are always use a corpus for this, but they also did something interesting. So, at that time, the main search engine was AltaVista. So, what they did they tried these queries over AltaVista. So, they said ok.

(Refer Slide Time: 13:31)

*Query search engine (AltaVista)*

- $P(\text{word})$  estimated by  $\text{hits}(\text{word})/N$
- $P(\text{word}_1, \text{word}_2)$  estimated by  $\text{hits}(\text{word}_1 \text{ NEAR } \text{word}_2)/N$

So, they have the estimate the three things probability of word probability of excellent, and probability of word and excellent occurring together. And they were using how many hits the word or the two words together are getting by the AltaVista engine that is again a nice method of computing the probabilities and co occurrence probability of this word. So, you give that a query find out how many hits the search engine is giving you. So, I have to assume a probability word, I see how many hit the word get by the engine divide by N. Probability word from word to again how many hits word 1 near word 2, some query like that get over the engine divide by N. Now, once I have these two measures, I can easily compute the PMI score.

(Refer Slide Time: 14:16)

Query search engine (Altavista)

- $P(\text{word})$  estimated by  $\text{hits}(\text{word})/N$
- $P(\text{word}_1, \text{word}_2)$  estimated by  $\text{hits}(\text{word}_1 \text{ NEAR } \text{word}_2)/N$

$\text{Polarity}(\text{phrase}) = \text{PMI}(\text{phrase, excellent}) - \text{PMI}(\text{phrase, poor})$

$$= \log_2 \left( \frac{\text{hits}(\text{phrase NEAR "excellent"}) \text{hits}("poor")}{\text{hits}(\text{phrase NEAR "poor"}) \text{hits}("excellent")} \right)$$

Presented by [REDACTED] Learning Outcome Examples Week 13: Text Mining

So, what is the polarity of the phrase that is PMI of the phrase with excellent minus PMI of the phrase with poor and that will give you this particular expression. If you want we can just quickly do that on paper.

(Refer Slide Time: 14:35)

positive  $w_1 \quad w_2 \quad \dots \quad w_n$

negative  $w_1 \quad w_2 \quad \dots \quad w_n$

$$\text{Polarity} = \text{PMI}(\text{phrase, excellent}) - \text{PMI}(\text{phrase, poor})$$
$$= \log_2 \left[ \frac{P(\text{phrase, excellent})}{P(\text{phrase, poor})} \frac{\text{hits}(\text{phrase})}{\text{hits}(\text{phrase})} \right]$$

$\downarrow \text{Probability of phrase}$   
 $\downarrow \text{Probability of phrase, poor}$   
 $\downarrow \text{Probability of phrase, excellent}$

So, we have to get PMI phrase excellent minus PMI phrase poor. So, how do I get that is equal to log probability phrase excellent divide by probability phrase probability excellent is. Minus log becomes you have to I men you are reversing it. So, it will become probability phrase poor probability phrase poor that will come out to be the PMI. And then

you can immediately say that this is repeating. So, you have only this one log probability phrase excellent, and this you obtain using hits phrase near excellent. Similarly, phrase near poor, this is hits for excellent, hits for poor and that is how you get this polarity of the phrase. And you do that for all the phrases that you have taken in your opinion lexicon you obtain some sort of polarity for this. So, these phrases occur with excellent a lot, these occur with poor a lot.

(Refer Slide Time: 16:00)

Example: A thumbs-up Review		
Phrase	POS tags	Polarity
online service	JJ NN	2.8
online experience	JJ NN	2.3
direct deposit	JJ NN	1.3
local branch	JJ NN	0.42
low fees	JJ NNS	0.33
true service	JJ NN	-0.73
other bank	JJ NN	-0.85
inconveniently located	JJ NN	-1.5
Average		0.32

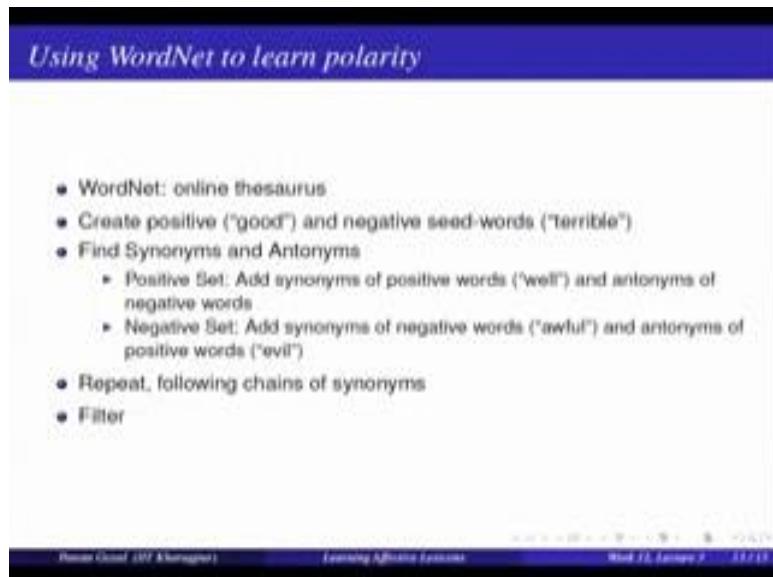


So, if you see from thumbs-up review, what are difference is that are they obtain, they obtained words like online service with this part of speech tags JJ NN polarity 2.8, online experience JJ NN 2.3, direct deposit JJ NN 1.3. And then they found out what is the average polarity score of each phrase that they got and they said something like 0.32. You can see there were some negative examples also there like low, true service but probably not negative, but inconveniently located first negative and they got a minus 1.5. So, they got some positive negative examples even in thumps up review, but the average that they obtained was positive 0.32.

Similarly they did for negative the thumbs-down reviews from the product site and while they were again something like direct deposits, online web, very handy with positive polarity they were many with negative polarity like virtual monopoly, lesser evil, other problems, low funds, unethical practices and they were you can see you have very, very negative score by their method and that was very nice. So, that was very interesting approach. So, by that they

were able to obtain such phrases and also tag them with the polarity values.

(Refer Slide Time: 17:00)



So, these are some algorithms there are many other algorithms literature, but one other method that you can always use is to use your WordNet. So, all of you know WordNet we talked about that in one of the earlier lectures. So, how do you use WordNet. So, again the idea is some sort of bootstrapping; in the WordNet find out some positive and negative sentiment words. Suppose, you find out this word is positive this word is negative. So, how do you bootstrap from there, how do you find more words? So, one simple thing that you can think is that find out all the synonyms of this word. So, all the synonyms would also can be given a positive polarity, if it is positive; if it is negative, they can be given a negative polarity this is one way.

Then you can find out the antonyms and give them the negative polarity; if it is positive it should be antonym should be given negative polarity; if it is negative and antonyms should be given positive polarity. And once you get more seed set, you can continue building over there. So, this can be one approach using WordNet. So, you create some positive that is good and negative seed words. So, positive is like good and negative like terrible. So, once you have the seed set you find the synonyms and antonyms.

So, wherever whatever are the synonyms of the positive word like good, you add to the positive set; and whatever the antonyms of the negative word like terrible you again add to the positive set, and you do the reverse for the negative set. So, add synonyms of

negative words and antonyms of positive word. So, like you will get well in the positive set, and awful and evil in the negative set.

And this you can further repeat now, you have got more words, you can keep on repeating them. And you might create some new set seed sets whenever you feel this is not giving me for that examples you might create some more seed sets and this might be again some simple nice method of build starting to bootstrapping your opinion lexicons, you can also combine different approaches together. So, you start some words from WordNet go to the other approach the first approach that we talked about using and, and but you can also start from their and then go to WordNet and to find an more synonyms, so both are possible.

And again there are many more methods, but I think this will be some interesting ideas that you can use for building your opinion lexicon. So, this was about this lecture. In the next lecture, what we will do we will see how you can also obtain the also see some nice trends about different words being used in the reviews, say review with rating 1 to 10 how are the different words, the positive and negative words are being used. And what are the nice ways in which you can use the sentiment lexicons that we have talked about. So, I will see you in the next lecture.

Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 64**  
**Computing with Affective Lexicons**

Welcome back for the lecture four of this week. So, in the last lecture, we discussed how do you bootstrap your opinion sentiment lexicons from using some simple seed sets, from your corpus and from the WordNet.

So, in this lecture, what we will see, so we will see two things mainly, suppose you have your sentiment lexicons, and you can use that directly to up to find out the sentiment score of your different sentences, and all in your data. So, these are very standard methods for doing that you can take average and all. But what are certain things that you have to keep in mind that is what we will talk about. And then from the different data sets we are use here abundance of sentiments like review data set over movies and all, what are that nice trends you see about different words.

So, what kind of major do you should use to be able to see those trends, also to be able to find out what words are getting what words are more prominent with positive and negative emotions in those.

(Refer Slide Time: 01:24)

*Learn word sentiment supervised by online review scores*

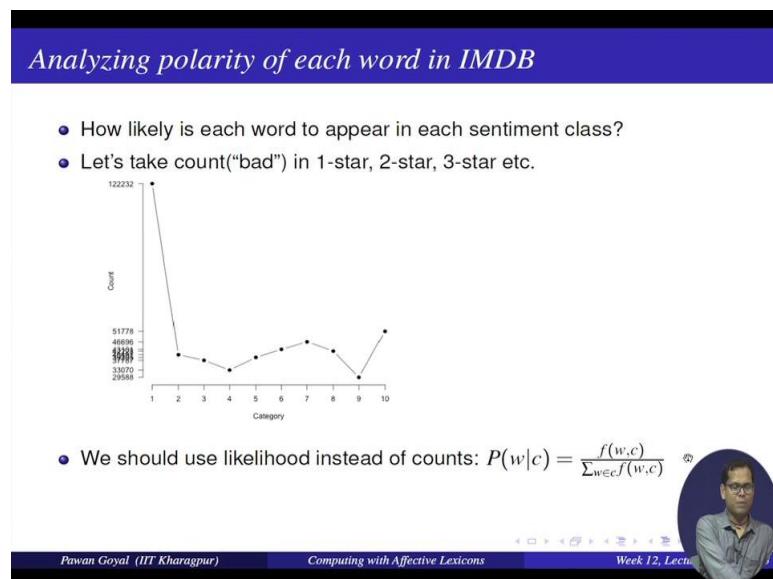
- Review datasets: IMDB, Goodreads, Amazon, Trip Advisor
- Each review has a score (1-5, 1-10 etc)
- Just count how many times each word occurs with each score (and normalize).

Pawan Goyal (IIT Kharagpur) Computing with Affective Lexicons Week 12, Lecture 4 2 / 13

So, let us see. Suppose, I want to learn what is the sentiment score of a word by see how often does that occur in a review corpus. So, I take some corpus from online corpus say I IMDB I take all the reviews from different movies. And now I am going to see how often how much correlated a particular word is with a particular rating a score. So, how do I do that? So, it can start with any review data sets.

So, here are some examples, you can take from movies, you can take IMDB, you want to take reviews for books, you can take good reads, you want to take reviews for hotels, and all you can take trip adviser and for various products you can take Amazon or other websites. Now, on different pages, you will find different ratings, somewhere you will find 1 to 5, somewhere you find 1 to 10 etcetera. And now I want to find out how often a word occurs in a particular sentiment class a particular rating class. So, what can be a good measure for doing that?

(Refer Slide Time: 02:34)



So, let us see. So, we are taking the examples from IMDB, and I want to find out I want to analyze polarity of each word in IMDB, so that is how often does that occur with different, different ratings. So, what will be the simplest measure that will come to your mind? You can see I have readings from 1 to 10, and I want to find out how often this word occurs with each of the rating 1, 2, 3, 4, 5, 6, 7, 8 times and so on and I can do like that k. So, let us count the word bad in 1 star, 2 stars, and 3 star etcetera. So, on x-axis, you can see different ratings 1 to 10; and on y-axis, you can see that count how many times a word occurs.

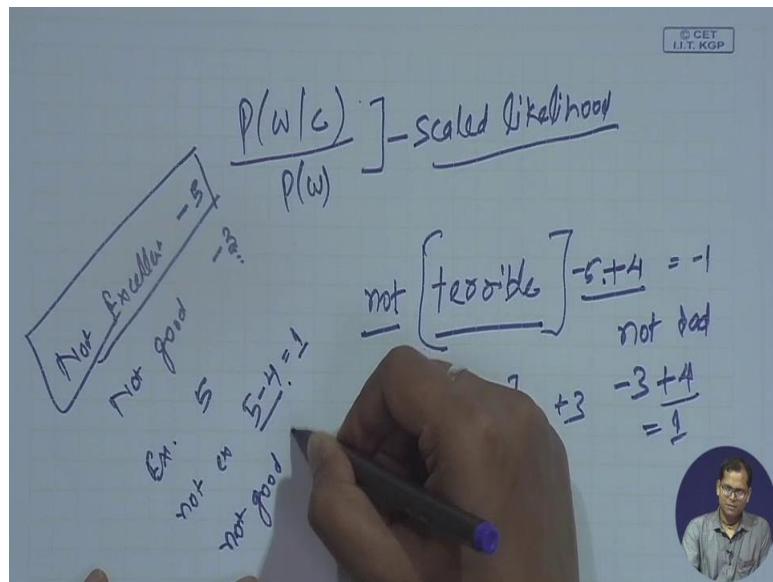
So, this can be a approach that can give you how many what are the counts of a word, but this

will not help you in finding out some sort of seeing a some sort of normalized picture. So, for example, does this word occur in one rating more with more probability than others other ratings, so what would be a criteria. So, you want to see that it might happen that in my corpus, there are more reviews with rating 1 than rating 5, so by that simple statistics the count of this word in 1 will be higher than count in 5. So, I should be able to do some normalization.

So, first normalization I can do is what is the count of the word in a particular rating divide by count of all the words in that rating to some sort of probability of a word in a particular rating that can be the first measure that I can take. So, what is the probability of a word given a particular rating? So, we will see is noting a rating and that you can obtain by seeing the number of times the word occurs with that rating divide by all the words that occur in that rating whatever times or you can say this is the size of that rating how many different words occur. So, this will give you the probability of the word occurring in that rating. So, this can help you normalize across the ratings. So, what is the probability of this word rating 1, rating 2, rating 3 and you can see does it have a higher probability rating one and so on.

But suppose now you want to compare across words. So, this is ok for comparing across the ratings, but if you want to compare across words this may not be a good measure. And why is that because it might again happen that this word is very, very common in lexicon so that means the probability of this word is actually very high. So, by that logic again the probability of this word occurring with this review might this rating might also be high. So, I want to do another normalization that takes into account what is the probability of this word.

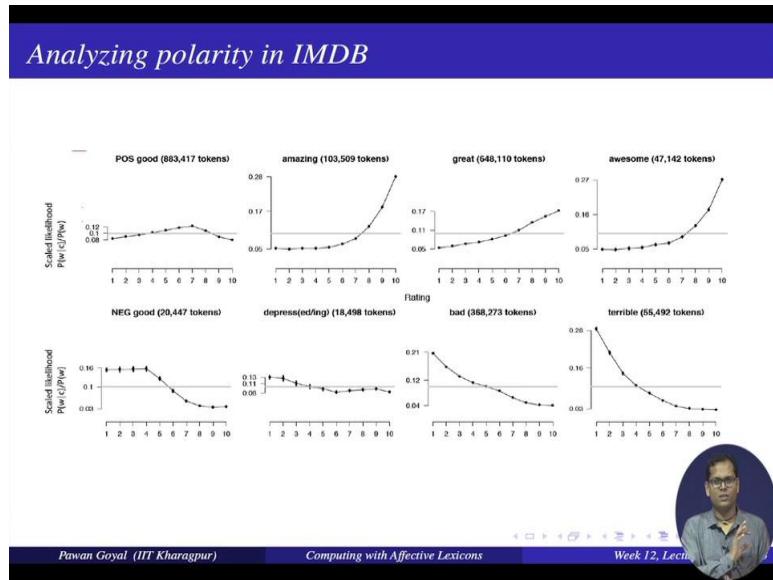
(Refer Slide Time: 05:28)



So, what can be an alternative measure. So, I already have probability of the word given this class, see there is a particular rating class. Now, I can divide it by the probability of the word is k itself and that will be a normalized measure. So, now it will be comparable across different words. So, I am seeing what is the probability of the word in the corpus divide in the particular rating divide by what is the probability of word overall. So, you can also think it like how much does that depart from its actual probability; if this is very different from its actual probability that means, yes, there is a very, very it is a nice indicative of a particular class.

So, this is called likelihood and this together is called as scaled likelihood. So, we will be trying to compare words across categories and different words by using their scaled likelihood. So, how do they occur in different ratings with its scaled likelihood? So, by that I am making them comparable across words by taking probability word given the class divide by probability word.

(Refer Slide Time: 06:45)



Now, let us take some example from IMDB data set. So, what we are seen on x-axis, we have different ratings 1 to 10; and on y-axis, we have the scaled likelihood. And it is a scale, so it will not go beyond 1, so it has to be between 0 to 1. So, what are you seeing here. So, let us see the first column positive good and negative good. So, what we are seeing as you go from rating 1 to 7 positive good is increasing so that means we are talking good without a negation it is increasing, but when you are from 7 to 10, it is decreasing. Negative good, it is high initially and then it is decreasing.

So, now can you make sense of that? So, when you are going to higher reviews initially good in increasing and then it is decreasing. And this is actually this might be the case why because yes when you go to in review of rating 1, you might not have much good return, but when you are going to higher review, yes, there is good. But when you go to even higher reviews, you might not be using a word like good, you might be using a better adjective like amazing and so on, fantastic.

So, you are using adjectives like that, so that is what you will see in the in the next columns. So, you take a adjective like amazing and you will see in reviews 1 to 5, it is nearly very low 0.05. And then start shooting up when it goes to 0.28 in the reviews of rating time that means, the word amazing occurs a lot in the reviews of rating time by the scaled likelihood and this give a nice picture.

Now, you talk about negative good adverse. So, adverse, decreasing, it was decreasing as you

go from high reviews. So, in review of 10 they want be much occurrence of not good. But now if you talk about terrible, or depressing; so, you have the word depressing depressive. So, you see that initially in rating 1, it was occurring with a high likelihood - the scaled likelihood; and as you go down, it keeps on decreasing. So, again this looks this pattern looks interesting.

Then you see the word like great it is not as steep as amazing, but again this is nice trend; it is starts increasing from 1 to 10, from 0.05 it goes up to 0.17 that similarly starts from 0.21 very high likelihood in reading one it goes up to 0.04 as you go to rating time. And similarly if you see awesome and terrible, they have similar trends as amazing. So, awesome is very similar to amazing, yes, occurs a lot in rating 10, you see the numbers also nearly same 0.28, 0.27.

And terrible is like very, very high in rating 1, 0.28 and goes to 0.03 at rating trend. So, this is how you can now compare different words that how likely do they occur with reviews of rating 1 2 and 10 and you can try to compare which words behave in a very similar manner. And you might even be able to use this method to find out by using a review corpus which words are actually positive, which words are negative. So, you can use it for the task that we discussed in the last lecture also.

(Refer Slide Time: 10:01)

The slide has a blue header bar with the title "Logical Negation". The main content area contains the following bullet points:

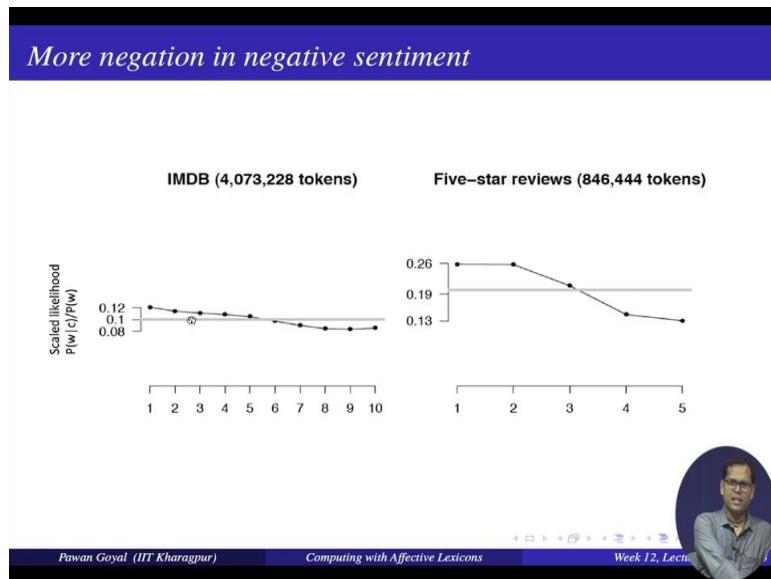
- Is logical negation (no, not) associated with negative sentiment?
- Potts experiment:
  - ▶ Count negation (not, n't, no, never) in online reviews
  - ▶ Regress against the review rating

At the bottom right of the slide, there is a circular video player showing a man with glasses and a grey shirt, presumably the speaker. The video player has a play button and other control icons. At the very bottom of the slide, there is a dark footer bar with the text "Pawan Goyal (IIT Kharagpur)", "Computing with Affective Lexicons", and "Week 12, Lecture 1".

Now, you can see some other trends also. So, how often, so negation is a big problem, so is it that whenever you had negation in the sentence does it always conveys a negative sentiment. So, is negative below is logical negation associate with the negative sentiment. So, Potts said

this experiment where they found out how many times negation like not, no, never were occurring in online reviews, and then they were trying to see its occurrence with the review ratings. So, how often do they occur with different review ratings?

(Refer Slide Time: 10:40)



Again they did took the scaled likelihood for IMDB and five-star reviews. And they found a very hyper sensitive trend here that in review of rating 1, it had a very high scaled likelihood 0.12. And as you go down up to 10, it had a 0.08, has not as different as you were seeing in terrible or awesome, but you can instant say that the negation is occurring more in the reviews with rating 1 than reviews in rating 10. And same thing they saw with other reviews were that were on five-star rating. So, it was occurring with 0.26 likelihood with review 1, review of rating 1 and a likelihood of 0.13 with the review of rating 5.

So, now, suppose I want to use this lexicon, I have a sentiment lexicon, I want to find use that to find out what is the sentimental score of a sentence in or a paragraph in my corpus. So, this will help I can just take the sentiment score of each of the words that is a very simple based on algorithm, you find out the sentiment of each word, add, take an average things like that and that will give you is score to the whole sentence. So, what I am trying to show here is that if you some linguistic intuition on top of that that might give you a better result.

So, for example, one particular problem with this is negation. So, what do I do, I have in my lexicon word like terrible and word like good; I know terrible is very bad like minus 5 and good is say plus 3. If I get a negation here, not terrible, not good, what do I do I add the

polarity of not with the polarity of terrible; similarly polarity of not with good that will not be a good approach. So, I should understand that these are some sort of function words they have some ten functions. So, can I take it as a function over this, a function over this? A simple function that you can think is just reverse the polarity whenever there is not reverse the polarity; terrible is minus 5, not terrible becomes plus 5, but is that a good approach, so that we will see.

Secondly, there are some other words that are also like function words. So, you can say so well word like very. So, you say good has some polarity, now you attach very to that very good what be the polarity and some words can be somewhat so then what will the polarity somewhat good. So, good has some polarity how do you give a polarity to the somewhat good. So, like that you can use some linguistic conditions to make much more science from your lexicon and avoid doing some mistakes with all these different function words.

(Refer Slide Time: 13:42)

Using Linguistic Intuitions

Using a sentiment lexicon also works.  
Some linguistic intuitions on top of that tends to give better results.

Pawan Goyal (IIT Kharagpur) Computing with Affective Lexicons Week 12, Lecture 4 7 / 13

So, you can use some linguistic conditions on top and that can give you some better results.

(Refer Slide Time: 13:47)

*Handling negation in simple addition of scores*

*Example words*

- Excellent +5
- good +3
- terrible -5
- bad -3

*Instead, a polarity shift works better*

- Not Excellent ( $5-4$ ) +1
- Not good ( $3-4$ ) -1
- Not terrible ( $-5+4$ ) -1
- Not bad ( $-3+4$ ) 1

Pawan Goyal (IIT Kharagpur) Computing with Affective Lexicons Week 12, Lecture 1

So, let us see. So, I have these words like excellent with plus 5, good with plus 3, terrible with minus 5, and bad with minus 3. So, one as we said one simple thing as we can do is when you have negation is that is just can just reverse the polarity. So, if you reverse the polarity that is what we get not excellent minus 5, not good minus 3, not terrible plus 5, and not bad plus 3. Now, just have a look at that for a second and see if that make sense. So, we are saying not excellent is minus 5 and not good is minus 3.

So, just think about it when I say excellent I mean this really good; when I say not excellent do I mean like terrible, if I say not excellent; that means, it is not excellent, but it may it is good, it is good, but not excellent. So, if I have to say it is not good, I will say not good. So, not excellent means something in the positive polarity. So, completely reversing the polarity will not be a good idea here. So, changing it from plus 5 to minus 5 will actually be a mistake.

Similarly, so not good is, but again you what you are doing not good is getting a better score than not excellent that is again not ideal. You want to give a lower score to not good than not excellent. Same thing you can think about to the other two words. So, you have not bad, not bad can mean something that is going towards good, but when you say not terrible you also you still mean that it is bad. So, not terrible should have a bad should have a negative score and much less than not bad, what is happening the reverse here. So, you seen not terrible is getting a higher score than not bad, this is not ideal.

So, you should have just reverse in the polarity, we can do something else and this is called you do something like a polarity shifting. So, if it is minus 5, you shift the polarity, so you add say plus 4 to bad. So, terrible is minus 5 add plus 4 you get minus 1, not terrible minus 5 minus 1 is ok. You do the same to the not bad. So, not bad will be minus 3; for bad plus 4 is equal to 1, so not bad starts getting good. So, you are getting a positive sense polarity.

Same thing you do with excellent, excellent is 5, and you say not excellent shift polarity shift to so you say 5 minus 4 here you were doing plus four here you are doing minus four this is just a number you can change this and this will give you 1. So, you still have a positive score. And not good, good has 3, you shift the polarity, you get minus 1, you get a negative polarity and that would be a much better approach than simply reversing the polarity. So, this is like linguistic intuition that you can use. So, here is the polarity shift and something like that some 5 minus 4 gives you plus 1, 3 minus 4 gives you minus 1 and you can see not good has a more negative sentiment than not excellent same you can see with not terrible and not bad.

(Refer Slide Time: 17:08)

## Handling Intensifiers

Intensifiers can be classified into two major categories,

- Amplifiers (e.g., very) increase the semantic intensity
- Downtoners (e.g., slightly) decrease it

*Rough values for some intensifiers*

Intensifier	Modifier (%)
slightly	-50
somewhat	-30
pretty	-10
really	+15
very	+25
extraordinarily	+50
(the) most	+100

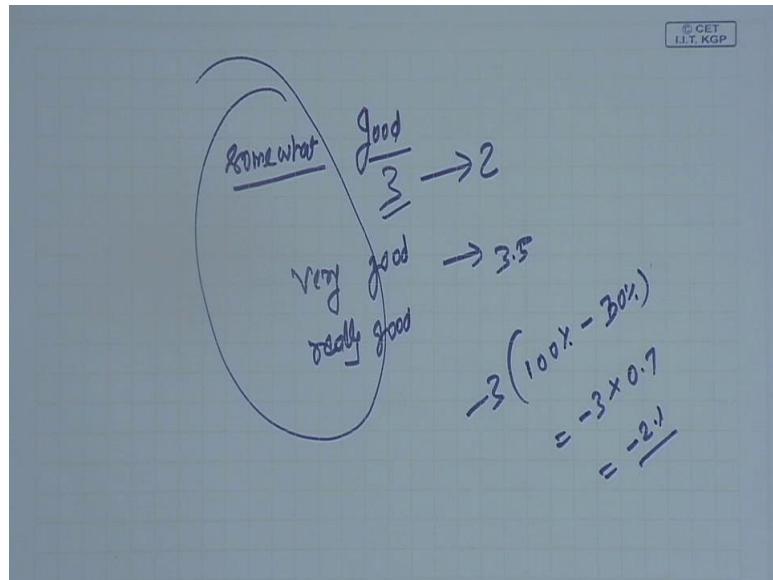
*Somewhat sleazy*

*sleazy: -3, somewhat sleazy:  $-3 \times (100\% - 30\%) = -2.1$*

Pawan Goyal (IIT Kharagpur) Computing with Affective Lexicons Week 12, Lecture 4 10 / 13

Similarly, how do you handle various intensifiers like very, somewhat etcetera? So, what do you see here, so you can have some amplifiers that can increase the intensity, and then there are certain downturners like slightly that slightly that can decrease it.

(Refer Slide Time: 17:28)



So, what do I want to do suppose you know the polarity for good, you know this score for good. Now, if it occurs with somewhat, somewhat good, suppose it is 3, you want to reduce this right somewhat good. So, you want to take from 3 to say 2. On the other hand, if it occurs with say very good or really good, so you want to increase that, so say 3.5 when we use 3, so very good might be 3.5.

So, again these are acting like some function words that can act as a function to modify your sentimental score of the main word. So, again in linguistic you can find outs and inform words in a paper. So, they had given some scores like slightly you do minus 50 percent, somewhat you do minus 30 percent, pretty minus 10 percent, really plus 15 percent. So, they are now an amplifier very is plus 25, extraordinary plus 50, and the most becomes plus 100.

So, you make these modifiers and how do you do computations. So, suppose you have a sentence like this somewhat sleazy. So, sleazy is minus 3. So, when you do somewhat sleazy, you will say ok, so what is this score of sleazy sorry somewhat minus 30 percent. So, you say minus 3, 100 percent minus 30 percent, so that will give you minus 3 into 0.7 it will give you minus 0.21, like that you can give a score with these downturners or amplifiers. So, this is again some linguistic intuitions that can be used.

(Refer Slide Time: 19:02)

*Irrealis moods: where the words may not be reliable*

- I thought this movie would be as good as the Grinch, but unfortunately, it wasn't.
- This should have been a great movie.

*What are the indicators?*

- conditional markers (*if*)
- negative polarity items like '*any*' and '*anything*'
- certain (mostly intensional) verbs (*expect, doubt*),
- questions
- words enclosed in quotes (which may be factual, but not necessarily reflective of the author's opinion)

Pawan Goyal (IIT Kharagpur) Computing with Affective Lexicons Week 12, Lecture 4 11 / 13

Then many times you have to be careful with specific sentences. So, like if people are using some irrealis moods. So, like I thought this movie would be as good as the Grinch, but unfortunately, it was not. So, here the author is saying I thought this movie would be as good as the Grinch. So, all the way there is a some positive words this say irrealis moods. So, you are saying I thought that this what happened, but this did not happen. So, this is again, so here you cannot just lie on your sentiment scores that are given in your lexicon.

Similarly you see a sentence like this, this should have been a great movie right. Again if you just use your sentiment lexicon you will say it is a positive polarity, but this is again a irrealis moods and you cannot rely on the words directly. So, you should have some way of finding out its irrealis moods or it will either probably change my polarity or I use some something else. So, specifically you need to be careful about using conditional markers if that was the case, then this would have been good then and so on.

Then something like any or anything, certain intentional words like I expected, I doubt whenever this occurs, you might not rely on what follows. Similarly, if there are questions in the sentence then also you should not be able to fully rely on the words. And then this is important many a times when you do the sentiment analysis over the corpus like news corpus. So, what do you see you will find various quotes? So, he said something, so now you cannot assign a sentiment to a sentence by whatever is there in the quotes because he just reporting some other sentiment, but actually the author did not have any sentiment. So, you are

reporting some sentiment.

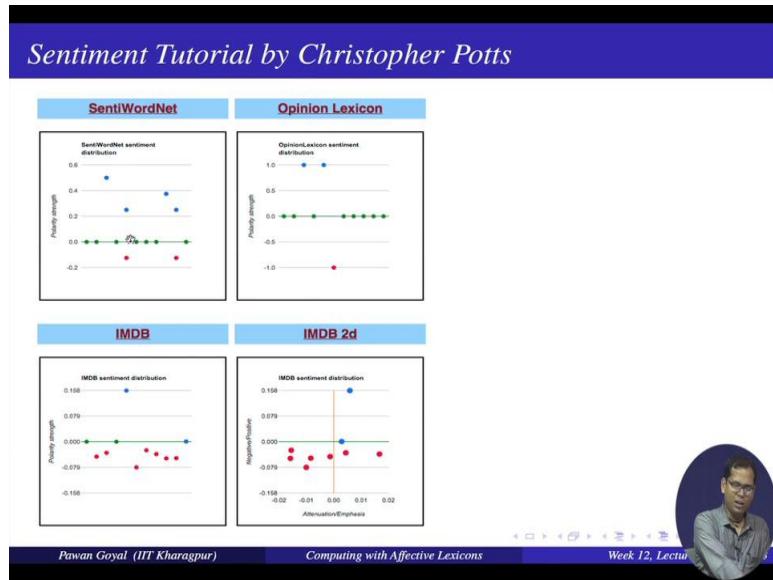
So, if you want to find out the sentiment of the sentence, it does not have any sentence. So, you should not use the sentiment of the quotes to give the sentiment of the sentence. So, the quotes also you should have to be careful with.

(Refer Slide Time: 21:03)

The screenshot shows a web page titled "Sentiment Tutorial by Christopher Potts". The main content area is titled "Text scoring" and contains the following text: "Shows how a variety of **sentiment lexicons** score novel texts. Such values could be used in many ways (as raw values, to derive percentages or ratios, as **classifier features**, ...).". Below this is a text input field with placeholder text "Enter your own text (max 140 characters), or" and a link "analyze a random tweet instead.". A "Submit" button is located below the input field. The next section is titled "Scores" and contains a text input field with the text "it sounds like a great plot but can't hold\_NEG up\_NEG .". At the bottom of the page, there is a navigation bar with links for previous and next slides, and a footer with the text "Pawan Goyal (IIT Kharagpur)", "Computing with Affective Lexicons", "Week 12, Lecture 4", and "12 / 13".

So, Christopher Potts has a very nice sentiment tutorial. So, you can just search for sentiment tutorial by Christopher Potts. And there you can also try different sentences and see how the tokenization is done. So, like it sounds like a great plot, but cannot hold up. So, you see the negation is coming with hold and up also. And further it tells you how different lexicons try to assign a score to this.

(Refer Slide Time: 21:28)



So, you will see how sentiwordnet deals with it, how opinion lexicon deals with it, how IMDB deals with it and so on. So, you see in sentiwordnet for different words, you find out what whether they are positive polarity or negative polarity. So, you can directly find out by using this tutorial website. So, and there are many other resources that will be helpful for you on that website itself. So, that was about this lecture that how do you compute with affective lexicons what kind of nice trends you can see about the words occurring in the review data sets.

So, we will end this week, and also the course in the next lecture. So, we will take one another application. So, you just try to give hints on and what is the aspect based sentiment analysis. So, till now what we are doing, we are giving a score to the sentence this is positive or negative sentiment. But suppose it is not like the whole I am saying positive about the whole hotel as such, I am writing hotel review, I may not be saying positive about the hotel or negative about the hotel, I might be saying about certain aspects of that may be the service was good, but the room may be the food quality was good, but the room was dirty.

So, I might be saying some positive about one aspect, but negative about the another aspect. So, there some simple methods of capturing those that would you will see in the next lecture.

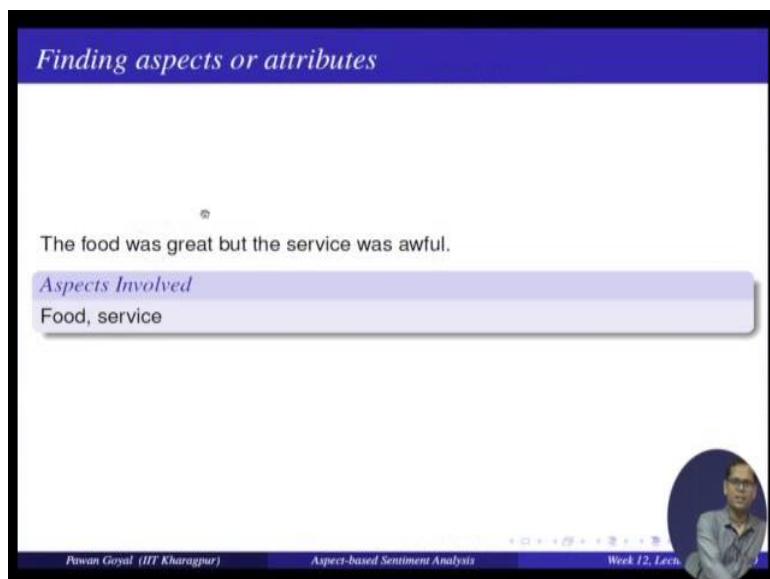
Thank you.

**Natural Language Processing**  
**Prof. Pawan Goyal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture - 65**  
**Aspect Based Sentiment Analysis**

Hello everyone, welcome to the final lecture of this week and also this course. So we will be finishing the topic of sentiment analysis. So, we discussed a lot about what is sentiment analysis, how do you obtain sentiment lexicons, how do you learn those how do you compute with those use some linguistic intuitions and so on. In this final lecture we will discuss few things like how do you use them some for some other applications like one of the application aspect base sentiment analysis. So, what do I mean by that.

(Refer Slide Time: 00:57)



The food was great but the service was awful.

Aspects Involved  
Food, service

So many of the times when we see the reviews they are not saying only positive only negative things about the product, about the hotel, about the movie, so you might have different sentiments about different aspects of a product ok. You can say I saw I got the camera and it was light weight so I am happy with that it was the price was ok it was not too expensive I am fine with that, but the picture quality is not good. So, you are not saying completely negative about the camera and neither you are saying completely positive about the camera, but if you can find out there are certain aspects and you can say to this aspect I am giving you this opinion to this aspect I am to this opinion that might be very very helpful.

Also so not only to get overall summary to what to what aspect this product is not doing well to what aspect is product is doing well and also try to you can use that to compare to products which product is better in terms of image quality, which product is better in terms of price, which term is better product is better in terms of say resolution and so on.

Now how do we capture that? So, again there are lot of lot of models that have been proposed for that we will see only some simple linguistic intuitions that can be applied for solving these problem, but there is a lot of literature around that. Let us take an example: I have this review the food was great, but the service was awful. So, you can see nearly 2 different polarities here the food was great and the service was awful.

Now as we have seen, sentiments have being expressed towards two different aspects see what are the aspects here, one is the food another is the service. So, can we capture that what are the different aspects that are being told in the reviews. Now how do we go about that? How do we find out for a particular domain what are the important aspects. So, again there are various algorithms. So, one very naive algorithm would be, we are taking that the sentences or reviews in a domain find out what kind of noun phrases or words occur a lot with some opinion words so; that means, a opinion word occurs just before them. That might be nice way of finding out what are the good aspects. Something's like this.

(Refer Slide Time: 03:22)

*Finding aspect/attribute/target of sentiment*

*Frequent phrases + rules*

- Find all highly frequent phrases across reviews ("fish tacos")
- Filter by rules like "occurs right after sentiment word"
- "... great fish tacos" means "fish tacos" a likely aspect

Casino	casino, buffet, pool, resort, beds
Children's Barber	haircut, job, experience, kids
Greek Restaurant	food, wine, service, appetizer, lamb
Department Store	selection, department, sales, shop, clothing



Pawan Goyal (IIT Kharagpur)      Aspect-based Sentiment Analysis      Week 12, Lecn.

So, I take all frequent phrases across the reviews like fish tacos and so on. Now among all that frequent phrases I find out those that occurs a lot right after some sentiment word. So the

idea is that if it is in aspect sometimes it so at least some time it will occur with the sentiment word mostly directly after it is a good and good fish taco and so on it is getting directly after sentiment word.

So whenever you see like great fish taco it means fish tacos is likely an aspect and you do that for all the frequent phrases that you have found in the corpus and I guess it is a very nice and simple intuition and by doing that itself you might get a lot of interesting aspects. You see like yes no. If you do it on a casino domain you find casino buffet, pool, resort, beds; children's barber you get like haircut, job, experience, kids; Greek restaurant- food, wine, service, appetizer, lamb; department store you will get words like selection, department, sales, shop, clothing, and now all of you all of them you see look like some nice aspects. You can further prove if you want by seeing; what are the aspects that occur explosively in this domain but not in other domain.

So, that will say these are the good aspects for these domain itself. So, that way you can find out what are the important aspects, once you found also aspects it will go through the corpus see; what are the opinions for each other aspects.

(Refer Slide Time: 04:53)

Extraction of aspect-opinion pairs		
RuleID	Observations	Examples
R1	$JJ \leftarrow amod \leftarrow NP$	The camera has a good screen. (good $\leftarrow$ amod $\leftarrow$ screen)
R2	$NP \rightarrow nsubj \rightarrow JJ$	The flash is brilliant. (flash $\leftarrow$ nsubj $\leftarrow$ brilliant)
R3	$VB \rightarrow dobj \rightarrow NP$	I love the image quality. (love $\rightarrow$ dobj $\rightarrow$ image quality)
R4	$NP \rightarrow nsubj \rightarrow JJ,$ $JJ \in \text{implicit aspect lexicon}$	The camera is expensive. (camera $\leftarrow$ nsubj $\leftarrow$ expensive)



Pawan Goyal (IIT Kharagpur)      Aspect-based Sentiment Analysis      Week 12, Lecture 1

Another possible approach for doing that is give some sorts of linguistic insights in how do people connects in opinion aspect in a sentence. Whenever you talk about some aspect within opinion is there some buffer structure that is followed and there has lot of rules that can be formulated by that and they are based on the dependency pass. How are they connected in the

dependency pass? If you see some examples here, there are four examples the camera has a good screen.

Now what you are seeing if you see dependency pairs there will be empty. That is a screen and they will be J J like good ok. Good will be adjective modify of a screen. Now this is a relation that can be captured from dependency pairs and suppose now you want to abstract that. Wherever a there is a J J and N P occurring in a dependency pairs such that J J is adjective modify and N P you can take this and say J J is your opinion and N P is your aspect.

And you can do that across all such occurrences that would be a nice rule and that can give you all possible aspect obedient pair that are occurring with this rule. Similarly here the flash is brilliant you are seeing the word flash occurs is as a subject of brilliant. Now you make a rule whenever N P is there and J J is it is and subject then you will take those aspect obedient pair. Similarly here I love the image quality, there is a verb and it is object is a non phrase. Verb log is the verb that is my opinion here and image quality in N P is my aspect.

Similarly here the camera is expensive you can find out camera N P and J J is the subject of that camera is expensive. So, here what is interesting is that expensive is not directly in an aspect it might be some implicit aspect it talks about an aspect like price. So by that you are saying by expensive. So, like that you can formulate some rules there are rules already available in the literature that can give you various aspect of obedient pairs from your review corpus. Suppose you have understood your aspects and you have some sentences where you know this is aspect 1 is aspect 2 aspects 3 they are all labeled.

Now once we have that then your task becomes very easy you can use a supervised classifier. What will you do? It will learn classify to classify whether the sentence talks about one of these aspects. It will learn from all these aspects the labeled sentences given a new sentence it will try to classify it and one of the aspects and, so that might be a nice approach.

(Refer Slide Time: 07:42)

If the aspects are well understood, use supervised classification.

**Rooms (3/5 stars, 41 comments)**

- (+) The room was clean and everything worked fine – even the water pressure ...
- (+) We went because of the free room and was pleasantly pleased ...
- (-) ...the worst hotel I had ever stayed at ...

**Service (3/5 stars, 31 comments)**

- (+) Upon checking out another couple was checking early due to a problem ...
- (+) Every single hotel staff member treated us great and answered every ...
- (-) The food is cold and the service gives new meaning to SLOW.

**Dining (3/5 stars, 18 comments)**

- (+) our favorite place to stay in biloxi.the food is great also the service ...
- (+) Offer of free buffet for joining the Play

Pawan Goyal (IIT Kharagpur)      Aspects-based Sentiment Analysis      Week 12, Lecture 5      5 / 10

So, like here, you are having different sentences one some sentences about rooms. So, the room was clean and everything work fine, we went because of the free room the worst hotel I had ever stayed at, even a stay shocking work room. Then something about service, upon checking out another couple were checking early due to a problem and everything single every single hotel staff treated us great about service, the food is cold and the service gives new meaning to slow again this is negative sentiment. But they are all occurring in their aspect of service.

Similarly what you will do, you will take different aspects and label difference sentences as per these aspects and then you use any supervised classifier like naive bayes or (Refer Time: 08:22) integration whatever and given a new sentence find out what is the aspect involved.

Now coming to some of the problems that you might face while doing that; so many a times we tend to think that a word will have only a particular sentiment scores either it is positive or it is negative, but this is not always the case you might have to build opinion lexicon corresponding to different aspects. So for one aspect this a word might have a positive meaning for another aspect it might have a negative meaning. This might very well depend on the particular aspect. So, let us take this example.

(Refer Slide Time: 09:02)

The slide has a blue header bar with the question "Do opinion phrases always have the same sentiment?". Below the header, there are two purple rectangular boxes. The first box contains the text "'Large' – positive or negative" and "Large screen vs. Large battery". The second box contains the text "'Long' – positive or negative" and "Long battery life vs. Long loading time". At the bottom of the slide, there is a dark footer bar with the text "Pawan Goyal (IIT Kharagpur)", "Aspect-based Sentiment Analysis", "Week 12, Lecture 5", and "6 / 10".

So, I have this word large, large by itself we might think it is positive, but it can be positive or negative so these are two examples. So, take large screen versus large battery. Large screen you might like right my phone has a larger screen that might be positive sentiment, but large battery might be like it is becoming very hobby it will be a negative sentiment. Similarly with long it can be a long battery life it is always good right, but if you say long loading time it might not be a good; so same word might have different sentiment with different aspects.

You might have talk about opinions specific to the aspects then there is another problem sometimes the aspects are mentioned explicitly. So, once you identify some aspects you can then find out whichever document they occur in whichever sentence they occur in you take that already, but sometimes they are mentioned only indirectly like the (Refer Time: 09:59) we saw. Price is a direct aspect, but if you say it was expensive it is indirect. So, it is again an interesting problem how do I find out what are the indirect aspects, implicit aspects.

(Refer Slide Time: 10:11)

The slide has a blue header bar with the title 'Explicit vs. Implicit Aspect Expressions'. The main content area contains a list of bullet points:

- *The picture quality of this camera is great* – 'picture quality' is an explicit aspect.
- *This camera is expensive* – 'expensive' is an implicit aspect expression describing 'price'.
- Implicit aspect expressions can be very complex as well, e.g., *This camera will not fit in a pocket* – "fit in a pocket" indicates the aspect 'size'.

At the bottom of the slide, there is footer text: 'Ptwan Goyal (IIT Kharagpur)', 'Aspect-based Sentiment Analysis', 'Week 12, Lecture 5', and '7/10'.

So, like here the picture quality of this camera is great, picture quality is an explicit aspect, but if I say the camera is expensive; expensive is an implicit aspect that corresponds to price. And sometimes then you have to be very very complex also ok. Like someone might say this camera does not fit in my pocket at all. So, saying will does not fill my pocket is saying it is very very this very huge size is huge. It is indicating the aspect size and that you are saying by some complex sentence.

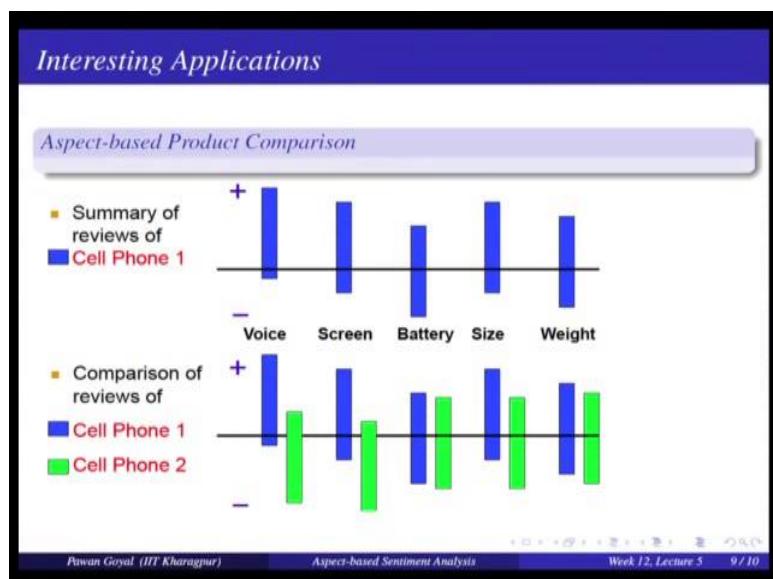
So, identify these that there is some implicit aspect together is also an interesting research problem. Once we have this aspect base sentiment analysis you can now use it for many many different applications. So, what will one nice application for that? So, think about a website like Amazon or Flipkart where different people are talking giving reviews about a particular product, it is one thing to say overall the review a positive or negative, but suppose you want to go further you can say about this product what are the interesting aspects like camera, the price, and the image quality and so on.

In each price what are the overall sentiment that is one thing, but suppose you can also say - what are people talking about price. What are people's comments about price, what are the people comments about image? So, there you want to gather all the sentences they talk about a particular aspect and then do some sort of summarization on top of that. That is why the summarizing methods that we talk in this course will be helpful. So, you say about the picture that is what people are saying, about the image quality that was that is what people are

seeing, about the size of the camera that is what people are saying. So, this field is called aspect based sentiment summarization, opinion summarization. We are taking opinions and trying to summarize it for each individual aspect. So, it will be like that.

So, you are taking reviews you are seeing these are the aspects like touch screen and voice quality within each you say these are the positive exam reviews is the negative reviews and then you summarize the positive and negative reviews separately for each of the aspect. Then another interesting application would be aspect based product comparison.

(Refer Slide Time: 12:28)



So, what are the different product comparison do you see in on the websites. So, the websites you see they are comparing products based on their specs. This has this much resolution and this is the size and so on. But they are not comparing on many interesting aspects like what is the voice quality of this mobile ok. They will not talk about that in those terms. So, by using the reviews of 2 products I can find out the common aspects and then try to compare them on those common aspects. Comparison can be again this has got this much score also I can talk about some summing.

So, this product p 1 is better than product p 2 in terms of this aspect because that is what customers are saying. And this will be a very nice application that you are able to say why should you prefer to go for product 1 than product 2 because when I read the review about this aspect that is what people are commenting and you are able to compare the 2 products. So, something like this. You can give summary reviews of one product like cell phone or you

can compare to cell phone 1 cell phone 2 and you can see in terms of voice this is more positive, this is more negative, screen this is more positive, this more negative, battery they are of the same size.

Cell phone 1 is more positive than cell phone 2. So, like that you can do now once you have taken the aspects you found out the aspects from the various reviews. And there are many many other things that you can do about sentiment. So, it is a huge topic. So that finishes our discussion on sentiment analysis and opinion extraction. As I was saying, we have only touched upon this field giving you basic intuitions, what are the simple methods you can use to at least get it started. And that is what we have done from many most of the application that we have talked about.

We talked about classifications, summarization, anterior linking, and information extraction so we did not cover them in very large details we only give you a intuitions and how simple things that you learn in the course can be helpful in solving these problems. And all of this I guess you would have got some nice idea.

(Refer Slide Time: 14:45)

*Conclusion*

*Many more NLP Applications*

- Machine Translation
- Question Answering
- Cross-lingual Applications
- Text Processing for social media, informal text

Deep Learning Techniques are being applied for most of the tasks.

Pawan Goyal (IIT Kharagpur)      Aspect-based Sentiment Analysis      Week 12, Lecture 5      10 / 10

But there are some many other application that we have not covered. One particular application for example, is machine translation. So, you take sentence from one language and you are trying to translate it to another language. Again so different techniques that we have used can be applied plus there are some other techniques that we will be used like, how do you align words in two languages and so on. Then there is nice application of question and

answering this is also becoming important. You have a lot of text in terms of Wikipedia and all and you guys asking a query and you do not just want irrelevant document, but you want an actual answer from those. So, this is also very interesting application, but it is very very challenging right now.

Then you can talk about various cross lingual application right. This is again very crucial right now, because you are having your information in one language like say English, but you are typing in Hindi. You are typing in a different language. So, by typing in Hindi how do you get information that is there in English? What kind of cross lingual method that you can use, again they will use various translation methods or you can use many other insights like using dictionaries that translate from one language to another language and so on. Again this is a nice application and then with a lot of social media coming in coming up applications on how do you process the language of social media.

So, how do you handle all these informal text? Yes; so how do you handle that, then how do you handle all this code switching and code mixing. What do I mean by that, when you write on twitters suppose you are a bilingual you use English as well as Hindi you start writing in English, suddenly you switch to Hindi. And you might mix words of English in Hindi in the same sentence. These are called code mixing and code switching depending on certain criteria. If I say code switching is one way here you change one language to another language completely at certain point and code mixing is where you keep on mixing words. So, it not directly completely switches one language to another.

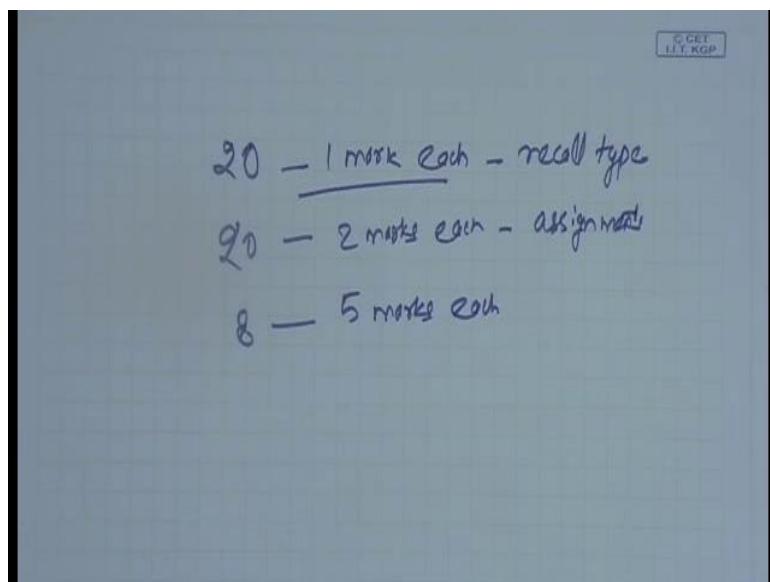
So, how do you process a language how do you find out what word belongs to what language how do you do all this part of speech tagging and say sentiment analysis over this kind of language is another interesting research direction that is coming up and there are many other application areas. Right now in many of this cases deep learning techniques are being applied, so we talk very briefly about this by word vectors, but many different deep learning methods are being applied, but that requires a complete different sort of course, where you are talk, you are thought about dual networks and this deep learning method.

But whatever techniques we have talked about in this course will be very very helpful when you would go and solve these sorts of problems and many of the application. And even when you are taking any application when you are start reading different research papers you will

again see that the basic the basic that I have covered in this course will be quiet helpful in understanding those research papers.

So, I think with that I will end this week and also this course. So, I hope that you found these 12 week in different course to be very very interesting, this you learnt a lot about this field of text mining and an NLP even the assignments probably helped you a lot in getting familiar with different tools and you feel confident, now that given a new problem that requires having some (Refer Time: 18:18) background you can try to tackle that. That was the main objective of this course and I hope that we have. So, you have been somewhat successful in that. So we will have an exam for this course, for that exam. Again the syllabus is the whole course that we have covered. All the slides will be there they want be any programming kind of questions in the exam. If I give you some idea, so this will be as per the format of NPTEL.

(Refer Slide Time: 18:52)



As per the format there will be first there will be 28 questions in the question paper. Dividing 20 20 8. So, 20 questions will be 1 mark each there will be like recall type questions. So, this will be coming from the slides. So, you should be able to recall certain things that we have talked about in the slide. Then 20 questions will be 2 marks each they are based on the assignments.

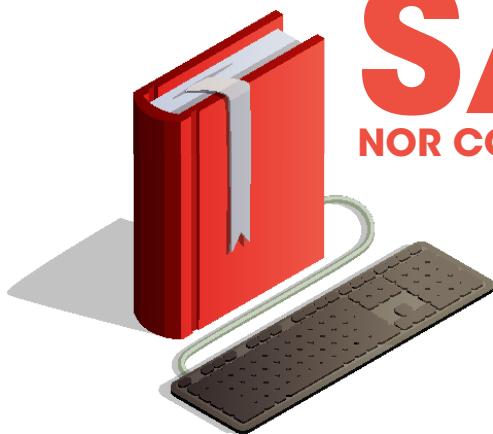
So, they will be coming based on different questions that you solved in your assignments, again there will there will not be programming kind of questions and then there will be 8 questions of 5 marks each these are some questions that were probably not covered in the

assignments and there will be some questions that we will require you to spend some more time, but hopefully you will be able to do that within that time that is allotted to you. So, one important thing that all these questions; most of these questions are like having only a single answer. So, they are multiple choice only a single answer is correct with 2 or 3 questions you might have to give the actual answer; so by in numerical as a numerical value.

So, this will happen with 2 or 3 or four questions. But if you have done this course, you should be able to solve to attempt all this problems and solve most of this. So, I will say, let me wish good luck for the further exam and thanks for being with us for the course and all my best wishes.

Thank you.

**THIS BOOK  
IS NOT FOR  
SALE  
NOR COMMERCIAL USE**



(044) 2257 5905/08



nptel.ac.in



swayam.gov.in