


## Regular Expressions

01\_Regular\_Expressions\_11-25.mp4 - VLC media player

Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



### Regular Expressions: Disjunctions

- Letters inside square brackets []

Pattern	Matches
[wW]oodchuck	Woodchuck, woodchuck
[1234567890]	Any digit


- Ranges [A-Z]

Pattern	Matches	
[A-Z]	An upper case letter	<u>D</u> renched Blossoms
[a-z]	A lower case letter	<u>m</u> y beans were impatient
[0-9]	A single digit	Chapter <u>1</u> : Down the Rabbit Hole

01:19 11:25

Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



### Regular Expressions: Negation in Disjunction

- Negations [^Ss]

  - Carat means negation only when first in []

Pattern	Matches	
[^A-Z]	Not an upper case letter	<u>O</u> yfn pripetchik
[^Ss]	Neither 'S' nor 's'	<u>I</u> have no exquisite reason"
[^e^]	Neither e nor ^	Look <u>h</u> ere
a^b	The pattern a carat b	Look up <u>a^b</u> now


02:48 11:25

## Note:

- “^” inside the bracket and outside the bracket acts differently
- Caret outside the bracket means start of the string

Media Playback Audio Video Subtitle Tools View Help


Dan Jurafsky



# Regular Expressions: More Disjunction

- Woodchucks is another name for groundhog!
- The pipe | for disjunction

Pattern	Matches
groundhog woodchuck	
yours mine	yours mine
a b c	= [abc]
[gG]roundhog [Ww]oodchuck	




04:04 11:25

- Pipe “|” stands for or


Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



# Regular Expressions: ? \* + .

Pattern	Matches	
colou?r	Optional previous char	<u>color</u> <u>colour</u>
oo*h!	0 or more of previous char	<u>oh!</u> <u>ooh!</u> <u>oooh!</u> <u>ooooh!</u>
o+h!	1 or more of previous char	<u>oh!</u> <u>ooh!</u> <u>oooh!</u> <u>ooooh!</u>
baa+		<u>baa</u> <u>baaa</u> <u>baaaa</u> <u>baaaaa</u>
beg.n		<u>begin</u> <u>begun</u> <u>begun</u> <u>beg3n</u>



Stephen C Kleene

Kleene \*, Kleene +


05:09 11:25

## Note:

- The character in the last example in place of "." can't be empty


Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



# Regular Expressions: Anchors ^ \$

Pattern	Matches
<code>^[A-Z]</code>	<u>P</u> alo Alto
<code>^[^A-Za-z]</code>	<u>1</u> "Hello"
<code>\.\$</code>	The end <u>.</u>
<code>.\$</code>	The end <u>?</u> The end <u>!</u>



06:25 11:25

- Caret stands for start of the string
- Dollar stands for end of string

## Character classes

.	any character except newline
\w \d \s	word, digit, whitespace
\W \D \S	not word, digit, whitespace
[abc]	any of a, b, or c
[^abc]	not a, b, or c
[a-g]	character between a & g

## Anchors

^abc\$	start / end of the string
\b	word boundary

## Escaped characters

\. \* \\	escaped special characters
\t \n \r	tab, linefeed, carriage return
\u00A9	unicode escaped ©

## Groups & Lookaround

(abc)	capture group
\1	backreference to group #1
(?:abc)	non-capturing group
(?=abc)	positive lookahead
(?!abc)	negative lookahead

## Quantifiers & Alternation

a* a+ a?	0 or more, 1 or more, 0 or 1
a{5} a{2,}	exactly five, two or more
a{1,3}	between one & three
a+? a{2,}?	match as few as possible
ab cd	match ab or cd

## Tokenization

- Fragments ⇒ main- mainly
- Filled pauses ⇒ uh



## How many words?

- I do uh main- mainly business data processing
  - Fragments, filled pauses
- Seuss's **cat** in the hat is different from other **cats**!
  - **Lemma**: same stem, part of speech, rough word sense
    - **cat** and **cats** = same lemma
  - **Wordform**: the full inflected surface form
    - **cat** and **cats** = different wordforms

01:21

14:25



## How many words?

$N$  = number of tokens

$V$  = vocabulary = set of types

$|V|$  is the size of the vocabulary

Church and Gale (1990):  $|V| > O(N^{1/2})$

	Tokens = $N$	Types = $ V $
Switchboard phone conversations	2.4 million	20 thousand
Shakespeare	884,000	31 thousand
Google N-grams	1 trillion	13 million

04:41

14:25

Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



## Issues in Tokenization


- Finland's capital → Finland Finlands Finland's ?
- what're, I'm, isn't → What are, I am, is not
- Hewlett-Packard → Hewlett Packard ?
- state-of-the-art → state of the art ?
- Lowercase → lower-case lowercase lower case ?
- San Francisco → one token or two?
- m.p.h., PhD. → ??

09:31 14:25

125%

Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



## Maximum Matching Word Segmentation Algorithm

- Given a wordlist of Chinese, and a string.
  - 1) Start a pointer at the beginning of the string
  - 2) Find the longest word in dictionary that matches the string starting at pointer
  - 3) Move the pointer over the word in string
  - 4) Go to 2

12:39 14:25

125%

Dan Jurafsky



## Max-match segmentation illustration

- Thecatinthehat
- Thetabledownthere
- Doesn't generally work in English!

the cat in the hat

the table down there

theta bled own there

14:12



14:25



This algorithm doesn't work for English

## **Normalization**





# Normalization

- Need to “normalize” terms
  - Information Retrieval: indexed text & query terms must have same form.
    - We want to match *U.S.A.* and *USA*
- We implicitly define equivalence classes of terms
  - e.g., deleting periods in a term
- Alternative: asymmetric expansion:
  - Enter: *window*                      Search: *window, windows*
  - Enter: *windows*                      Search: *Windows, windows, window*
  - Enter: *Windows*                      Search: *Windows*
- Potentially more powerful, but less efficient

01:07



11:47



Generally it is done to make SEARCHING easy

## Case Folding





## Case folding

- Applications like IR: reduce all letters to lower case
  - Since users tend to use lower case
  - Possible exception: upper case in mid-sentence?
    - e.g., *General Motors*
    - *Fed* vs. *fed*
    - *SAIL* vs. *sail*
- For sentiment analysis, MT, Information extraction
  - Case is helpful (*US* versus *us* is important)

01:56



11:47



## Lemmatization

Dan Jurafsky



## Lemmatization

- Reduce inflections or variant forms to base form
  - *am, are, is* → *be*
  - *car, cars, car's, cars'* → *car*
- *the boy's cars are different colors* → *the boy car be different color*
- Lemmatization: have to find correct dictionary headword form
- Machine translation
  - Spanish **quiero** ('I want'), **quieres** ('you want') same lemma as **querer** 'want'

02:50

11:47

## Morphology

Dan Jurafsky



## Morphology

- **Morphemes:**
  - The small meaningful units that make up words
  - **Stems:** The core meaning-bearing units
  - **Affixes:** Bits and pieces that adhere to stems
    - Often with grammatical functions

03:37

11:47


For example:

In the string “Stems” - “stem” is a stem and “s” is an affix

## Stemming

Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



# Stemming

- Reduce terms to their stems in information retrieval
- *Stemming* is crude chopping of affixes
  - language dependent
  - e.g., ***automate(s), automatic, automation*** all reduced to ***automat.***

*for example compressed and compression are both accepted as equivalent to compress.*

→

for exampl compress and compress ar both accept as equal to compress

04:57 11:47 125%

Stemming is a simplified version of Lemmatization

## Porter's Algorithm



### Step 1a

### Step 1b

### Step 2 (for long stems)

### Step 3 (for longer stems)

07:10

11:47



## 33


11:47



# Sentence Segmentation

Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



## Sentence Segmentation

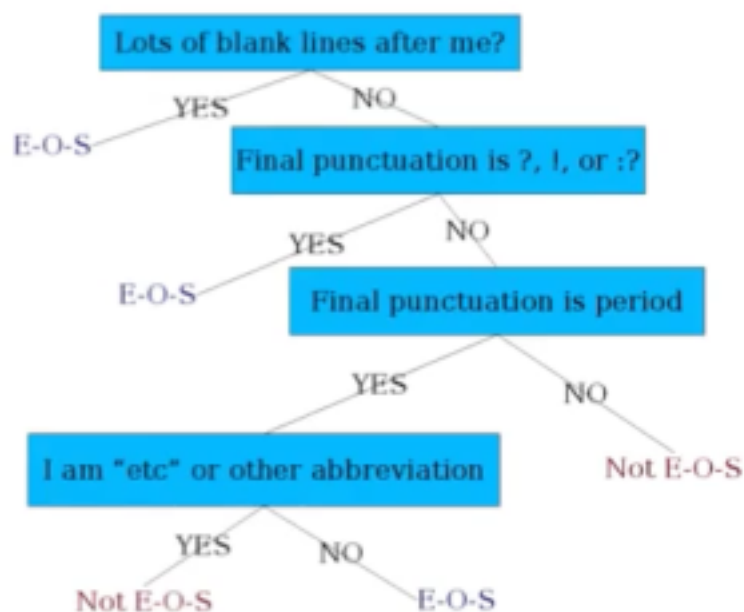
- !, ? are relatively unambiguous
- Period "." is quite ambiguous
  - Sentence boundary
  - Abbreviations like Inc. or Dr.
  - Numbers like .02% or 4.3
- Build a binary classifier
  - Looks at a "."
  - Decides EndOfSentence/NotEndOfSentence
  - Classifiers: hand-written rules, regular expressions, or machine-learning

01:00 05:34

125%



## Determining if a word is end-of-sentence: a Decision Tree



01:05

05:34



## More sophisticated decision tree features

- 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-s)
  - Probability(word after "." occurs at beginning-of-s)


02:33

05:34

## Minimum Edit Distance

Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



# Edit Distance

- The minimum edit distance between two strings
- Is the minimum number of editing operations
  - Insertion
  - Deletion
  - Substitution
- Needed to transform one into the other

01:11 07:04

125%





## Defining Min Edit Distance (Levenshtein)

- Initialization

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

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

- Recurrence Relation:

For each  $i = 1 \dots M$

For each  $j = 1 \dots N$

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

- Termination:

$D(N, M)$  is distance

Minimum Edit Distance

		a	b	c	d	e	f
	o	1	2	3	4	5	6
a	1	0	1	2	3	4	5
z	2	1	1	2	3	4	5
c	3	2	2	1	2	3	4
e	4	3	3	2	2	2	3
d	5	4	4	3	2	3	3

Handwritten annotations on the right side of the table:

- abc<sup>x</sup>def
- a → z, c → e, d → d
- f → d
- d → x
- b → z

### Note:

- If the comparing characters are different then the value will be minimum of (left value or top value or left diagonal value) plus 1
- if the comparing characters are same then the value will be equal to the left diagonal value

## **Language Modelling**

WHY?



## Probabilistic Language Models

- Today's goal: assign a probability to a sentence

- Machine Translation:

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

- Spell Correction

- The office is about fifteen **minuets** from my house

- $P(\text{about fifteen minutes from}) > P(\text{about fifteen minuets from})$

- Speech Recognition

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

- + Summarization, question-answering, etc., etc.!!

Why?

01:02

08:41



## Probabilistic Language Modeling

- Goal: compute the probability of a sentence or sequence of words:

$$P(W) = P(w_1, w_2, w_3, w_4, w_5 \dots w_n)$$

- Related task: probability of an upcoming word:

$$P(w_5 | w_1, w_2, w_3, w_4)$$

$$P(w_1, w_2, w_3, w_4, w_5)$$

- A model that computes either of these:

$P(W)$  or  $P(w_n | w_1, w_2 \dots w_{n-1})$  is called a **language model**.

01:50

08:41


Compute the Joint Probability of a sentence or a sequence of words

$P(\text{its, water, is, so, transparent, that})$

This can be calculated using **Chain Rule of Probability**

Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



## Reminder: The Chain Rule

- Recall the definition of conditional probabilities
$$P(A|B) = \frac{P(A,B)}{P(B)}$$
Rewriting: 
$$P(A|B)P(B) = P(A,B)$$
$$P(A,B) = P(A|B)P(B)$$
- More variables:
$$P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)$$
- The Chain Rule in General
$$P(x_1, x_2, x_3, \dots, x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2) \dots P(x_n|x_1, \dots, x_{n-1})$$

03:27 08:41

100%



# The Chain Rule applied to compute joint probability of words in sentence

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

$P(\text{"its water is so transparent"}) =$

$$P(\text{its}) \times P(\text{water} | \text{its}) \times P(\text{is} | \text{its water}) \\ \times P(\text{so} | \text{its water is}) \times P(\text{transparent} | \text{its water is so})$$

04:09

08:41

How to calculate this probability?



## How to estimate these probabilities

- Could we just count and divide?

$$P(\text{the | its water is so transparent that}) = \\ \frac{\text{Count}(\text{its water is so transparent that the})}{\text{Count}(\text{its water is so transparent that})}$$

- No! Too many possible sentences!
- We'll never see enough data for estimating these

04:57


08:41

Markov Assumption is used to calculate this probability


## ***Markov Assumption***

Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



# Markov Assumption



Andrei Markov

- Simplifying assumption:  
 $P(\text{the l its water is so transparent that}) \approx P(\text{the l that})$
- Or maybe  
 $P(\text{the l its water is so transparent that}) \approx P(\text{the l transparent that})$

05:37 08:41

100%



## Markov Assumption



Andrei Markov

- Simplifying assumption:

$$P(\text{the l its water is so transparent that}) \approx P(\text{the l that})$$

- Or maybe

$$P(\text{the l its water is so transparent that}) \approx P(\text{the l transparent that})$$

05:34



08:41



## Markov Assumption

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

- In other words, we approximate each component in the product

$$P(w_i | w_1 w_2 \dots w_{i-1}) \approx P(w_i | w_{i-k} \dots w_{i-1})$$

05:39



08:41







## Simplest case: Unigram model

$$P(w_1 w_2 \dots w_n) \approx \prod_i P(w_i)$$

Some automatically generated sentences from a unigram model

fifth, an, of, futures, the, an, incorporated, a,  
a, the, inflation, most, dollars, quarter, in, is,  
mass

thrift, did, eighty, said, hard, 'm, july, bullish

that, or, limited, the

06:15



08:41



In Unigram Model, the words are independent of each other. So the sentence apparently doesn't make sense.



## Bigram model

- Condition on the previous word:

$$P(\underline{w_i} \mid \underline{w_1 w_2 \dots w_{i-1}}) \approx P(\underline{w_i} \mid \underline{w_{i-1}})$$

texaco, rose, one, in, this, issue, is, pursuing, growth, in,  
a, boiler, house, said, mr., gurria, mexico, 's, motion,  
control, proposal, without, permission, from, five, hundred,  
fifty, five, yen

outside, new, car, parking, lot, of, the, agreement, reached

this, would, be, a, record, november

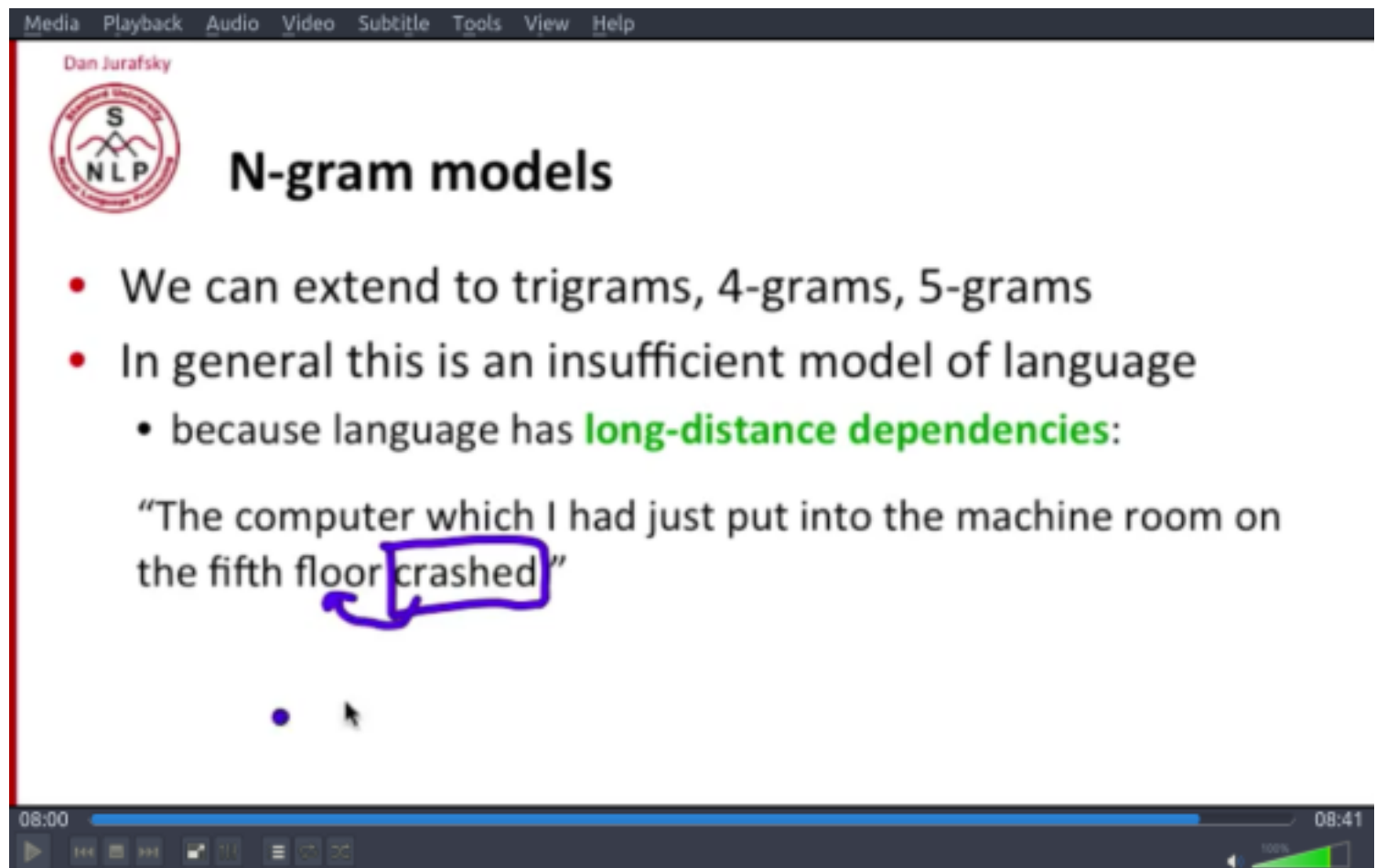
07:02



08:41




In Bigram Model the word is dependent on its previous word.



Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



## N-gram models

- We can extend to trigrams, 4-grams, 5-grams
- In general this is an insufficient model of language
  - because language has **long-distance dependencies**:

"The computer which I had just put into the machine room on the fifth floor crashed"

08:00 08:41

Like in the above example, N-gram model can fail with sentences having long distance dependencies.

Because the word "crashed" is not related to floor. Here the actual subject is computer. Hence the N-gram model will fail here.

But more or less it works OK for most of the sentences.



## An example

### Estimating Bigram Probabilities

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

<s> I am Sam </s>

<s> Sam I am </s>

<s> I do not like green eggs and ham </s>

$$P(I | <s>) = \frac{2}{3} = .67$$

$$P(</s> | Sam) = \frac{1}{2} = 0.5$$

$$P(Sam | <s>) = \frac{1}{3} = .33$$

$$P(Sam | am) = \frac{1}{2} = .5$$

$$P(am | I) = \frac{2}{3} = .67$$

$$P(do | I) = \frac{1}{3} = .33$$

$$\frac{\text{Count} ( am, Sam )}{\text{Count} ( am )} = \frac{1}{2}$$

02:05

09:38



## Practical Issues

- We do everything in log space
  - Avoid underflow Multiplying Numbers like 0.002 \* 0.00001
  - (also adding is faster than multiplying)

$$p_1 \times p_2 \times p_3 \times p_4 = \log p_1 + \log p_2 + \log p_3 + \log p_4$$

07:19

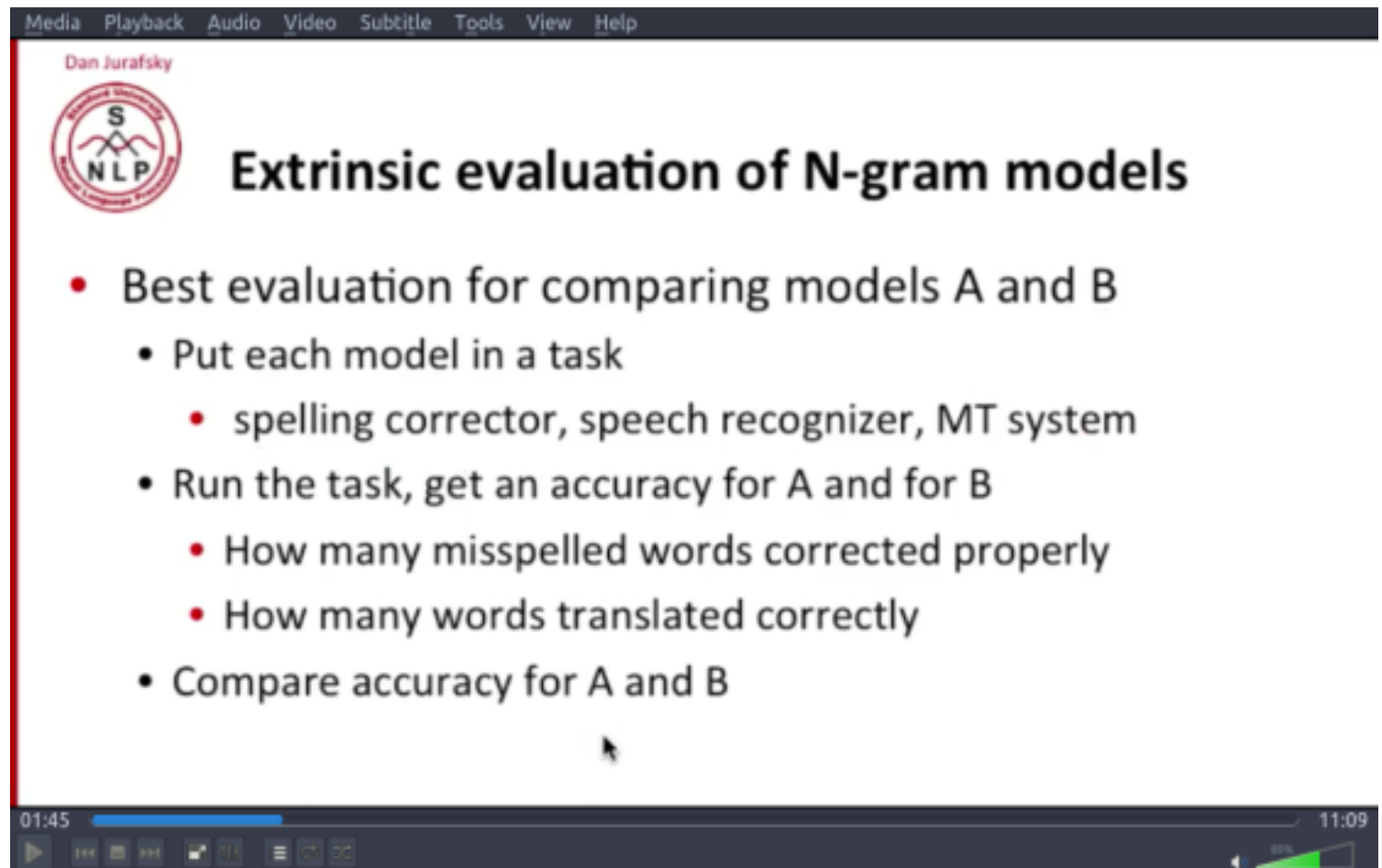
09:38

# Evaluation of Model

Model can be evaluated in two ways

1. Extrinsic
2. Intrinsic

Extrinsic evaluation is done by using external data and comparing the accuracy. Like using the unseen test data and checking the accuracy on it.



The screenshot shows a video player interface. At the top, there is a menu bar with options: Media, Playback, Audio, Video, Subtitle, Tools, View, and Help. Below the menu bar, the video content is displayed. It features a red circular logo on the left with the letters 'S' and 'NLP' inside, and the text 'Dan Jurafsky' above it. To the right of the logo, the title 'Extrinsic evaluation of N-gram models' is written in a large, bold, black font. Below the title, there is a bulleted list of points:


- Best evaluation for comparing models A and B
  - Put each model in a task
    - spelling corrector, speech recognizer, MT system
  - Run the task, get an accuracy for A and for B
    - How many misspelled words corrected properly
    - How many words translated correctly
- Compare accuracy for A and B

At the bottom of the video player, there is a progress bar showing the current time as 01:45 and the total time as 11:09. There are also various control icons like play, pause, stop, and volume.

Intrinsic evaluation is the evaluation of the model itself rather than any external application

Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



# Difficulty of extrinsic (in-vivo) evaluation of N-gram models

- Extrinsic evaluation
  - Time-consuming; can take days or weeks
- So
  - Sometimes use **intrinsic** evaluation: perplexity
  - Bad approximation
    - unless the test data looks **just** like the training data
    - So **generally only useful in pilot experiments**

02:40 11:09

A type of intrinsic evaluation is Perplexity

Perplexity is the probability of the test set, normalized by the number of words.



# Perplexity

The best language model is one that best predicts an unseen test set

- Gives the highest  $P(\text{sentence})$

Perplexity is the probability of the test set, normalized by the number of words:

Chain rule:

For bigrams:

$$\begin{aligned} \text{PP}(W) &= \frac{P(w_1 w_2 \dots w_N)^{-\frac{1}{N}}}{1} \\ &= \sqrt[N]{\frac{1}{P(w_1 w_2 \dots w_N)}} \\ \text{PP}(W) &= \sqrt[N]{\prod_{i=1}^N \frac{1}{P(w_i | w_1 \dots w_{i-1})}} \\ \text{PP}(W) &= \sqrt[N]{\prod_{i=1}^N \frac{1}{P(w_i | w_{i-1})}} \end{aligned}$$

Minimizing perplexity is the same as maximizing probability

06:56



11:09



## The Shannon Game intuition for perplexity

- From Josh Goodman
- How hard is the task of recognizing digits '0,1,2,3,4,5,6,7,8,9'
  - Perplexity 10
- How hard is recognizing (30,000) names at Microsoft.
  - Perplexity = 30,000
- If a system has to recognize
  - Operator (1 in 4)
  - Sales (1 in 4)
  - Technical Support (1 in 4)
  - 30,000 names (1 in 120,000 each)
  - Perplexity is 54
- Perplexity is weighted equivalent branching factor

average  
branching  
factor

The perplexity is actually 53, not 54

08:45



11:09



## Perplexity as branching factor

- Let's suppose a sentence consisting of random digits
- What is the perplexity of this sentence according to a model that assign  $P=1/10$  to each digit?

$$\begin{aligned} \text{PP}(W) &= P(w_1 w_2 \dots w_N)^{-\frac{1}{N}} \\ &= \left(\frac{1}{10}\right)^{-\frac{1}{N}} \\ &= \frac{1}{10}^{-1} \\ &= 10 \end{aligned}$$

$$\begin{aligned} &P(w_1, w_2, \dots, w_N)^{-\frac{1}{N}} \\ &P\left(\frac{1}{10} \cdot \frac{1}{10} \cdot \frac{1}{10} \cdot \frac{1}{10}\right)^{-\frac{1}{4}} \\ &\left(\frac{1}{10}\right)^{-\frac{1}{4} \cdot 4} \end{aligned}$$

10:03



11:09



## Lower perplexity = better model

- Training 38 million words, test 1.5 million words, WSJ

N-gram Order	Unigram	Bigram	Trigram
Perplexity	962	170	109

10:29



11:09






# Generalization

Media Playback Audio Video Subtitle Tools View Help

Dan Jurafsky



## The perils of overfitting

- N-grams only work well for word prediction if the test corpus looks like the training corpus
  - In real life, it often doesn't
  - We need to train robust models that generalize!
  - One kind of generalization: Zeros!
    - Things that don't ever occur in the training set
      - But occur in the test set

04:01 05:15

60%



# Zeros

- Training set:
  - ... denied the allegations
  - ... denied the reports
  - ... denied the claims
  - ... denied the request
- Test set
  - ... denied the offer
  - ... denied the loan

$$P(\text{"offer"} \mid \text{denied the}) = 0$$

04:56



05:15



## ***Zero Probability Problem***

When the sentences in the test set have never appeared in the training set, the probability assigned to it is zero.  
This produces Zero Probability Problem.

Dan Jurafsky



## Zero probability bigrams

- Bigrams with zero probability
  - mean that we will assign 0 probability to the test set!
- And hence we cannot compute perplexity (can't divide by 0)!

05:11



05:15



### ***Smoothing (Add One Estimate)***

To avoid Zero Probability Problem, Smoothing or Add One Estimate technique is used.

Dan Jurafsky



## The intuition of smoothing (from Dan Klein)

- When we have sparse statistics:

$P(w \mid \text{denied the})$

3 allegations ✓

2 reports ✓✓

1 claims ✓✓✓

1 request ✓✓✓

7 total

- Steal probability mass to generalize better

$P(w \mid \text{denied the})$

2.5 allegations

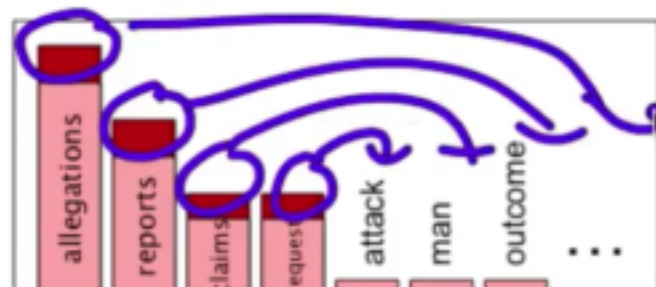
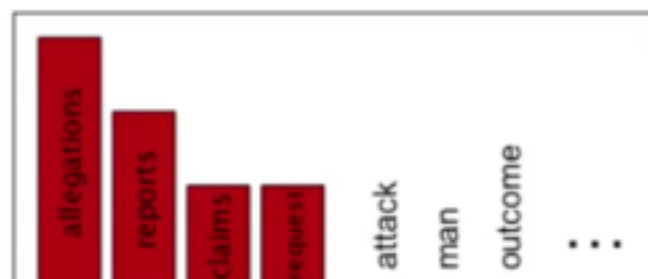
1.5 reports

0.5 claims

0.5 request

2 other

7 total



01:04

06:30

Dan Jurafsky



## Add-one estimation

- Also called Laplace smoothing
- Pretend we saw each word one more time than we did
- Just add one to all the counts!

$$P_{MLE}(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

- MLE estimate:

$$P_{Add-1}(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i) + 1}{c(w_{i-1}) + V}$$

- Add-1 estimate:

02:06

06:30



## Add-1 estimation is a blunt instrument

- So add-1 isn't used for N-grams:
  - We'll see better methods
- But add-1 is used to smooth other NLP models
  - For text classification
  - In domains where the number of zeros isn't so huge.

06:28



06:30

