Information Retrieval (IN4325)

Syntax

Dr. Nava Tintarev
Assistant Professor, TU Delft

TU Delft

Admin

- NLP project proposal: due **Friday 21st**
- Interviews are also on theory
- Remember office hours 9 -11.30 am on Fri, on https://queue.tudelft.nl/
- P1: Fokkens, Antske, et al. "Offspring from reproduction problems: What replication failure teaches us." Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vol. 1. 2013

Project proposal

Problem description: which problem will you tackle and what is interesting about the problem? If you reproduce a paper make sure to reference the original paper.
Resources: which data and tools will you be using
Methodology: if you choose to reproduce a paper, tell us which part of the paper (if not all) you will reproduce; if you choose your own research idea, tell us what type of algorithm/approach you are proposing and what your baseline(s) will be
Background readings: list at least 5 related papers that you will read to add context to your research. Reputable venues. NO ARXIV please!
Evaluation: how will you evaluate your algorithm/approach? Which evaluation metrics will you use? Need to be suitable for this task.

Submission: PDF, 1 per group. On Brightspace, should contain the group name and **group members** (name, student IDs).
Feedback: the course team will provide feedback on the project proposal within a few days.

TU Delft

Last time ...

- The function of Natural Language Processing
- NLP applications
- Typical natural processing tasks
- Typical components/sub-tasks

Some terms

- Normalization
- Stemming
- Lemmatization
- Segmentation
- Tokenization
- Stopwords

Today you will learn about **Syntax**

Syntax - comes from the greek *sýntaxis (σύνταξις) (coordination)*

σύν *syn*, "together", and τάξις *táxis*, "an ordering"

Refers to the way words are arranged together.

Ambiguity makes NLP hard: “Crash blossoms”

Teacher Strikes Idle Kids

Hospitals Are Sued by 7 Foot Doctors

Juvenile Court to Try Shooting Defendant

Applications: Sentiment Analysis

Great processor, but the memory sucks

This vacuum cleaner really sucks

Sentiment Analysis

- Simplest task, classification:
  - Is the attitude of this text positive or negative?
- More complex, regression:
  - Rank the attitude of this text from 1 to 5
- Advanced:
  - Detect the target, source, or complex attitude types

Sentiment Analysis

- Sentiment analysis is the detection of **attitudes**

  "enduring, affectively colored beliefs, dispositions towards objects or persons"

  1. **Holder (source)** of attitude
  2. **Target (aspect)** of attitude
  3. **Type** of attitude
     - From a set of types
       - *Like, love, hate, value, desire,* etc.
     - Or (more commonly) simple weighted **polarity**:
       - *positive, negative, neutral,* together with *strength*
  4. **Text** containing the attitude
     - Phrase, sentence, or entire document

10

Named Entity Recognition

In 1917, Einstein applied the general theory of relativity to model the large-scale structure of the universe. He was visiting the United States when Adolf Hitler came to power in 1933 and did not go back to Germany, where he had been a professor at the Berlin Academy of Sciences. He settled in the U.S., becoming an American citizen in 1940. On the eve of World War II, he endorsed a letter to President Franklin D. Roosevelt alerting him to the potential development of "extremely powerful bombs of a new type" and recommending that the U.S. begin similar research. This eventually led to what would become the Manhattan Project. Einstein supported defending the Allied forces, but largely denounced using the new discovery of nuclear fission as a weapon. Later, with the British philosopher Bertrand Russell, Einstein signed the Russell-Einstein Manifesto, which highlighted the danger of nuclear weapons. Einstein was affiliated with the Institute for Advanced Study in Princeton, New Jersey, until his death in 1955.

Tag colours:

LOCATION TIME PERSON ORGANIZATION MONEY PERCENT DATE

Syntax

- **Part-of-speech (POS) tagging**
  - Deep parsing
  - Shallow parsing
- **Language models**
  - Markov Chains: N-grams
  - Smoothing
  - Interpolation
  - Backoff
  - Web-Scale language models
- **Applications**
  - Named-entity recognition
  - Sentiment analysis

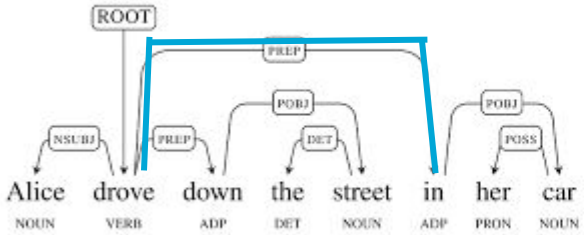
TU Delft

The Parts of Speech

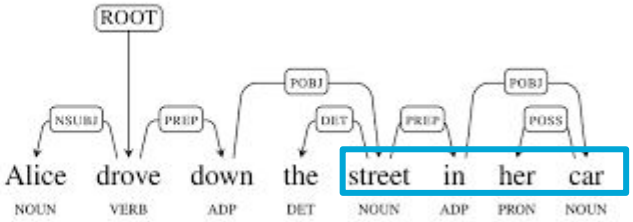
Every name is called a **noun**,
As *field* and *fountain*, *street* and town.
In place of noun the **pronoun** stands,
As *he* and *she* can clap their hands.
The **adjective** describes a thing,
As *magic* wand or *bridal* ring.
The **verb** means action, something done,
To *read* and *write*, to *jump* and *run*.
How things are done the **adverbs** tell,
As *quickly*, *slowly*, *badly*, *well*.
The **preposition** shows relation,
As *in* the street or *at* the station.
Conjuntions join, in many ways,
Sentences, words, *or* phrase *and* phrase.
The **interjection** cries out, "*Hark!*
I need an exclamation mark!"

Through Poetry, we learn how each
of these make up **THE PARTS OF SPEECH**.

Part-of-speech tagging



Alice is driving in her car



Or the street is located in her car.

The ambiguity arises because the preposition *in* can either modify *drove* or *street*; this example is an instance of what is called *prepositional phrase attachment ambiguity*.

Part-of-speech tagging

Helps with

- Stemming
- Select particular information e.g., nouns, named entities
- Parsing
- Word sense disambiguation (e.g., **interest**)

Tagsets

- Penn Treebank (45)
- Brown corpus (87)
- Dutch training data in NLTK, e.g., CONLL 2002 (conll2002)

POS, word classes, morphological classes, or lexical tags

Part-of-speech tagging

```
The/DT grand/JJ jury/NN comment/VBD on/IN a/DT
number/NN of/IN other/JJ topics/NNS.
```

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*	WP$	possessive wh-	*whose*
NNS	noun, plural	*llamas*	WRB	wh-adverb	*how, where*
NNP	proper noun, singular	*IBM*	$	dollar sign	*$*
NNPS	proper noun, plural	*Carolinas*	#	pound sign	*#*
PDT	predeterminer	*all, both*	"	left quote	' or "
POS	possessive ending	*'s*	"	right quote	' or "
PRP	personal pronoun	*I, you, he*	(	left parenthesis	[, (, {, <
PRP$	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*			

Part-of-speech tagging

Ambiguity:

Book/VB that/DT flight/NN./.

Book/NN that/DT flight/NN./.

Rule-based tagging

Statistical methods: Markov chains and n-grams

Transformation-base tagging and memory-tagging

Syntax

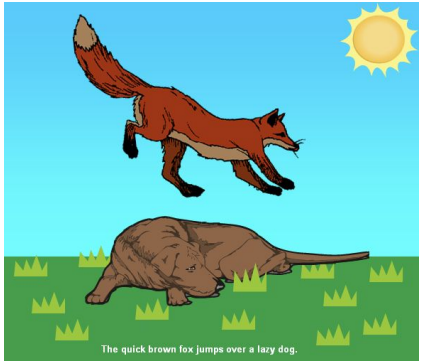
Part 1:

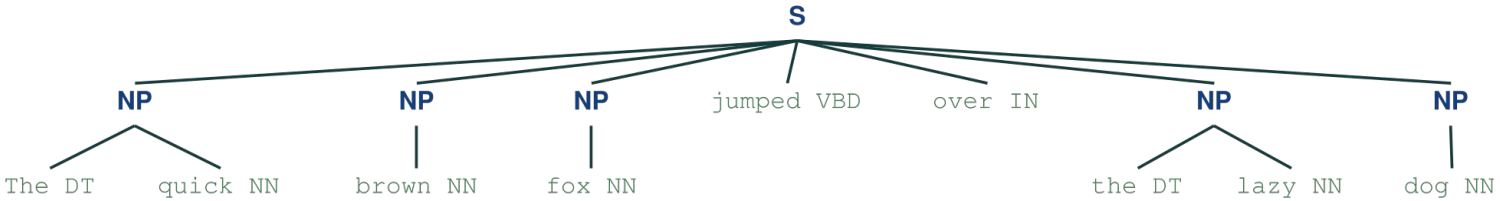
- Part-of-speech (POS)
- **Parsing**
- Language modeling

Part 2:

- Sentiment analysis
- Named-entity recognition

Parsing


The quick brown fox jumps over a lazy dog.


S
NP
NP
NP
jumped VBD
over IN
NP
NP
The DT
quick NN
brown NN
fox NN
the DT
lazy NN
dog NN

Parser

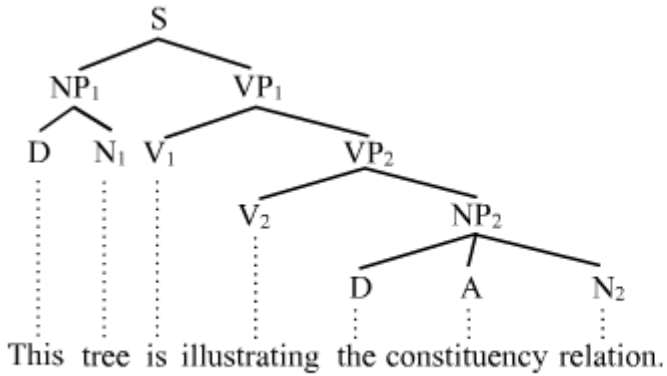
- **Input:** Text
- **Output:** Linguistic structure

- Can be for PoS tagging, disambiguation etc

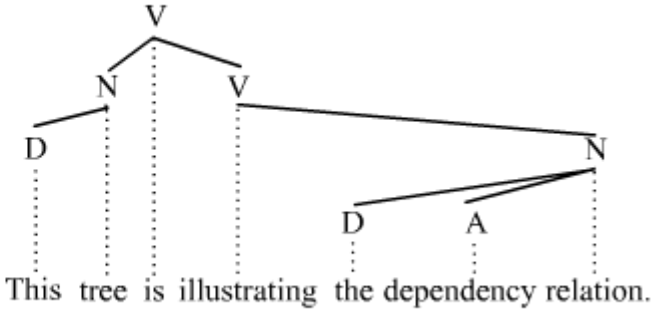
- Dependency grammars versus phrase structure grammars
- Deep parsing versus shallow parsing

Grammars

- **Phrase structure** (context-free/constituency) grammars
  - Constituency is one-to-one-or-more (label-element) corresp.
  - More information
- **Dependency grammar**
  - one-to-one correspondence element/constituent, less ambiguous
  - dependency grammars are **word** (or morph) grammars.
  - Dependency structures are more minimal, much fewer nodes.
-



Phrase structure



Dependency

Parsing versus Chunking

- Partial parse = shallow parse
- Identify segments that are likely to contain valuable information.

Chunking

- Identify major PoS types: nouns, verbs, adjectives, prepositional phrases
- Used e.g., for named entity recognition
- No hierarchical structure
- Type of sequential classification

Chunking example: IOB tagging

Beginning (**B**), internal (**I**), outside (**O**) of a chunk:

The	morning	flight	from	Denver	has	arrived.
B_NP	**I_NP**	**I_NP**	**B_PP**	**B_NP**	**B_VP**	**I_VP**

Only NP tags gives:

The	morning	flight	from	Denver	has	arrived.
B_NP	**I_NP**	**I_NP**	**O**	**B_NP**	**O**	**O**

Syntax

Part 1:

- Part-of-speech (POS)
- Parsing
- **Language modeling**

Part 2:

- Sentiment analysis
- Named-entity recognition

Probabilistic Language Models

Why?

Assign a probability to a sentence

- Machine Translation:
  - P(**high** winds tonite) > P(**large** winds tonite)
- Spelling Correction
  - The office is about fifteen **minutes** from my house
    - P(about fifteen **minutes** from) > P(about fifteen **minutse** from)
- Speech Recognition
  - P(I saw a van) >> P(eyes awe of an)
- + Summarization, question-answering, etc., etc.!!

TU Delft

Probabilistic Language Modeling

- Goal: compute the probability of a sentence or sequence of words:
  - $P(W) = P(w_1,w_2,w_3,w_4,w_5 \ldots w_n)$
- Related task: probability of an upcoming word:
  - $P(w_5|w_1,w_2,w_3,w_4)$
- A model that computes either of these:
  - $P(W)$ or $P(w_n|w_1,w_2 \ldots w_{n-1})$ is called a **language model**.

- Better: **the grammar**
- But **language model** or **LM** is standard

Clarification - language model

- language model in NLP is not the same as in search
- in search when we talk about LMs
  - in search there is a query
  - here there is no query
- different granularity
  - in search the LM is based on a document
  - here the LM is based on a corpus
- Both are probabilistic methods, used very differently
  - in search, likelihood of generating the query from the document
  - Here, likelihood of words (co)-occurring (useful as feature for classifiers, next word prediction)

Maximum Likelihood Estimates

- The maximum likelihood estimate
  - of some parameter of a model M from a training set T
  - maximizes the likelihood of the training set T given the model M
- Suppose the word "banana" occurs 400 times in a corpus of a million words
- What is the probability that a random word from some other text will be "banana"?
- MLE estimate is 400/1,000,000 = .0004
- This may be a bad estimate for some other corpus
  - But it is the **estimate** that makes it **most likely** that "banana" will occur 400 times in a million word corpus.

How to compute P(W)

- How to compute this joint probability:
  - P(its, water, is, so, transparent, that)
- **Intuition:** let's rely on the Chain Rule of Probability

The Chain Rule

- Conditional probability: P(B|A)
- 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
- $P(x_1,x_2,x_3,\ldots,x_n) = P(x_1)\, P(x_2|x_1)\, P(x_3|x_1,x_2) \ldots P(x_n|x_1,\ldots,x_{n-1})$

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

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

P("its water is so transparent") =

P(its) × P(water|its) × P(is|its water)

× P(so|its water is) × P(transparent|its water is so)

How to estimate these probabilities

- Could we just count and divide?

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

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

TU Delft

Andrei Markov

Markov Assumption

- Simplifying assumption:

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

- Or maybe

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

Markov Assumption

Chain Rule →

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

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

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

TU Delft

Simplest case: Unigram model

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

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
```

Bigram model

- Condition on the previous word:

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

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

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."

- However, we can often get away with N-gram models.

- The generativity of language (never said before) is a problem that affects deep-learning too!

Estimating bigram probabilities

The Maximum Likelihood Estimate

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

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

An example

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

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

$P(\texttt{I} \mid \texttt{<s>}) = \frac{2}{3} = .67$ $P(\texttt{Sam} \mid \texttt{<s>}) = \frac{1}{3} = .33$ $P(\texttt{am} \mid \texttt{I}) = \frac{2}{3} = .67$

$P(\texttt{</s>} \mid \texttt{Sam}) = \frac{1}{2} = 0.5$ $P(\texttt{Sam} \mid \texttt{am}) = \frac{1}{2} = .5$ $P(\texttt{do} \mid \texttt{I}) = \frac{1}{3} = .33$

Bigram estimates of sentence probabilities

P(<s> I want english food </s>) =

P(I|<s>)
× P(want|I)
× P(english|want)
× P(food|english)
× P(</s>|food)
= .000031

Practical Issues

We do everything in log space

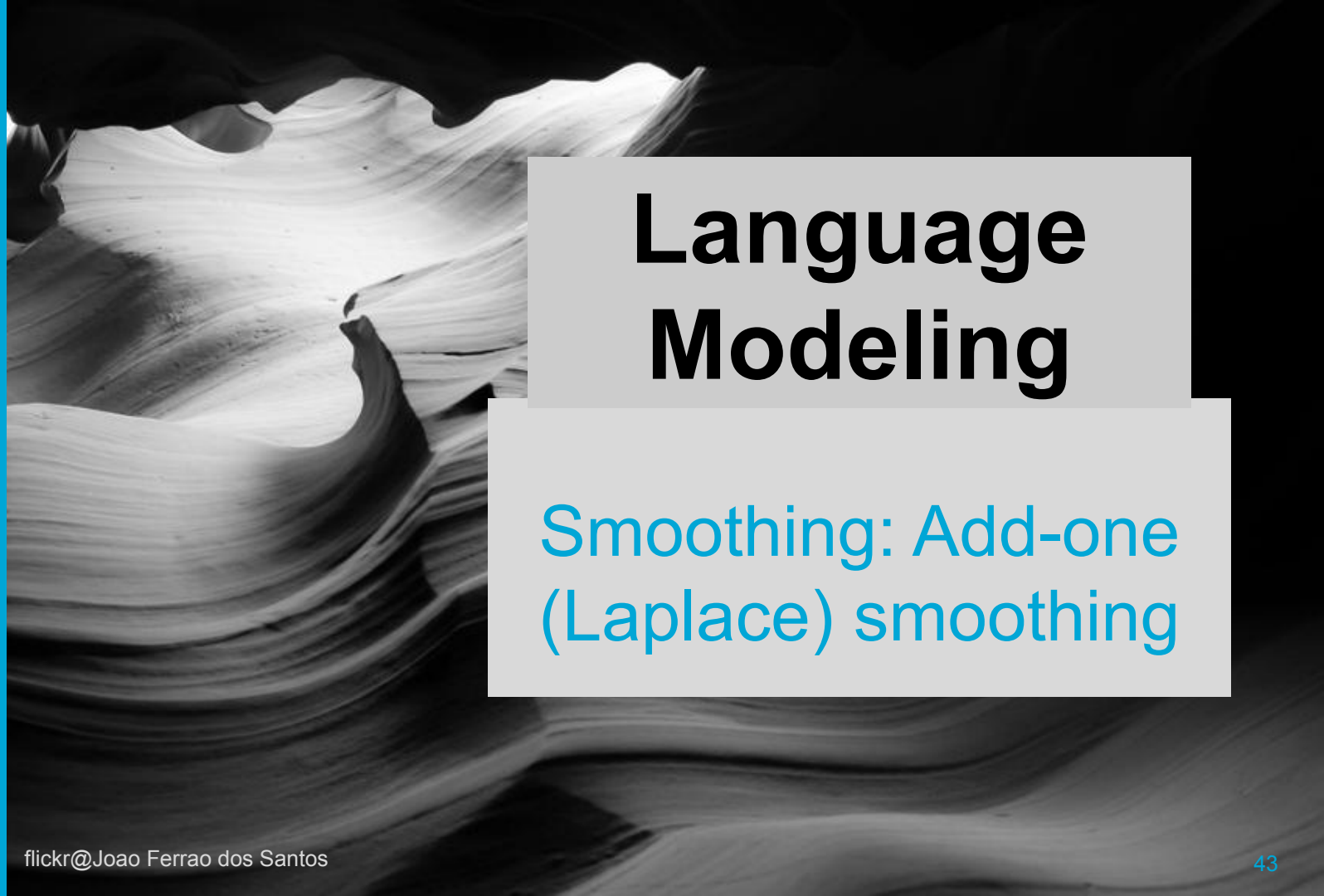
- Avoid underflow
- (also adding is faster than multiplying)

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

Questions?

Language Modeling

Smoothing: Add-one (Laplace) smoothing

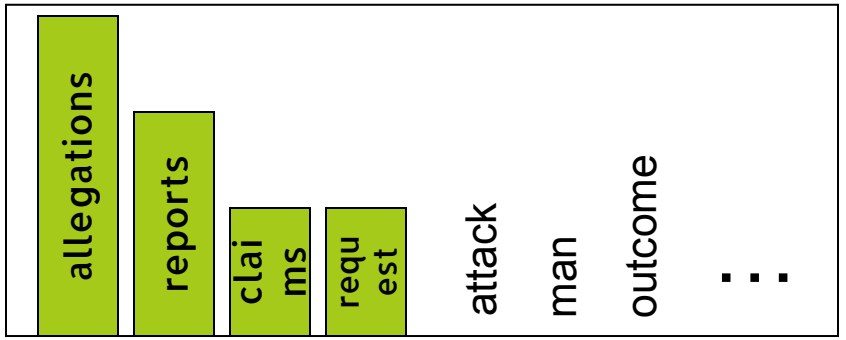


flickr@Joao Ferrao dos Santos

The intuition of smoothing (from Dan Klein)

When we have sparse statistics:

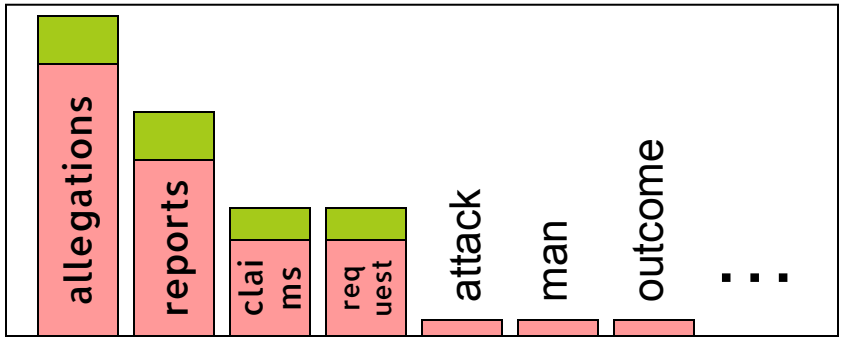
P(w | denied the)
- 3 allegations
- 2 reports
- 1 claims
- 1 request
- **7** total



`Steal' probability mass to generalize better

P(w | denied the)
- 2.5 allegations
- 1.5 reports
- 0.5 claims
- 0.5 request
- 2 other

7 total



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!

- MLE estimate:

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

- Add-1 estimate:

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

Berkeley Restaurant Corpus:

- Bigram counts for 8 words (V=1446)
- 9332 sentences
- Zeros in blue

	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

Berkeley Restaurant Corpus: Laplace smoothed bigram counts

	i	want	to	eat	chinese	food	lunch	spend
i	6	828	1	10	1	1	1	3
want	3	1	609	2	7	7	6	2
to	3	1	5	687	3	1	7	212
eat	1	1	3	1	17	3	43	1
chinese	2	1	1	1	1	83	2	1
food	16	1	16	1	2	5	1	1
lunch	3	1	1	1	1	2	1	1
spend	2	1	2	1	1	1	1	1

Laplace-smoothed bigrams

$$P^*(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)+1}{C(w_{n-1})+V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
spend	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058

Reconstituted counts

Multiply probability with original count for comparison

$$c^*(w_{n-1}w_n) = \frac{[C(w_{n-1}w_n)+1] \times C(w_{n-1})}{C(w_{n-1})+V}$$

	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

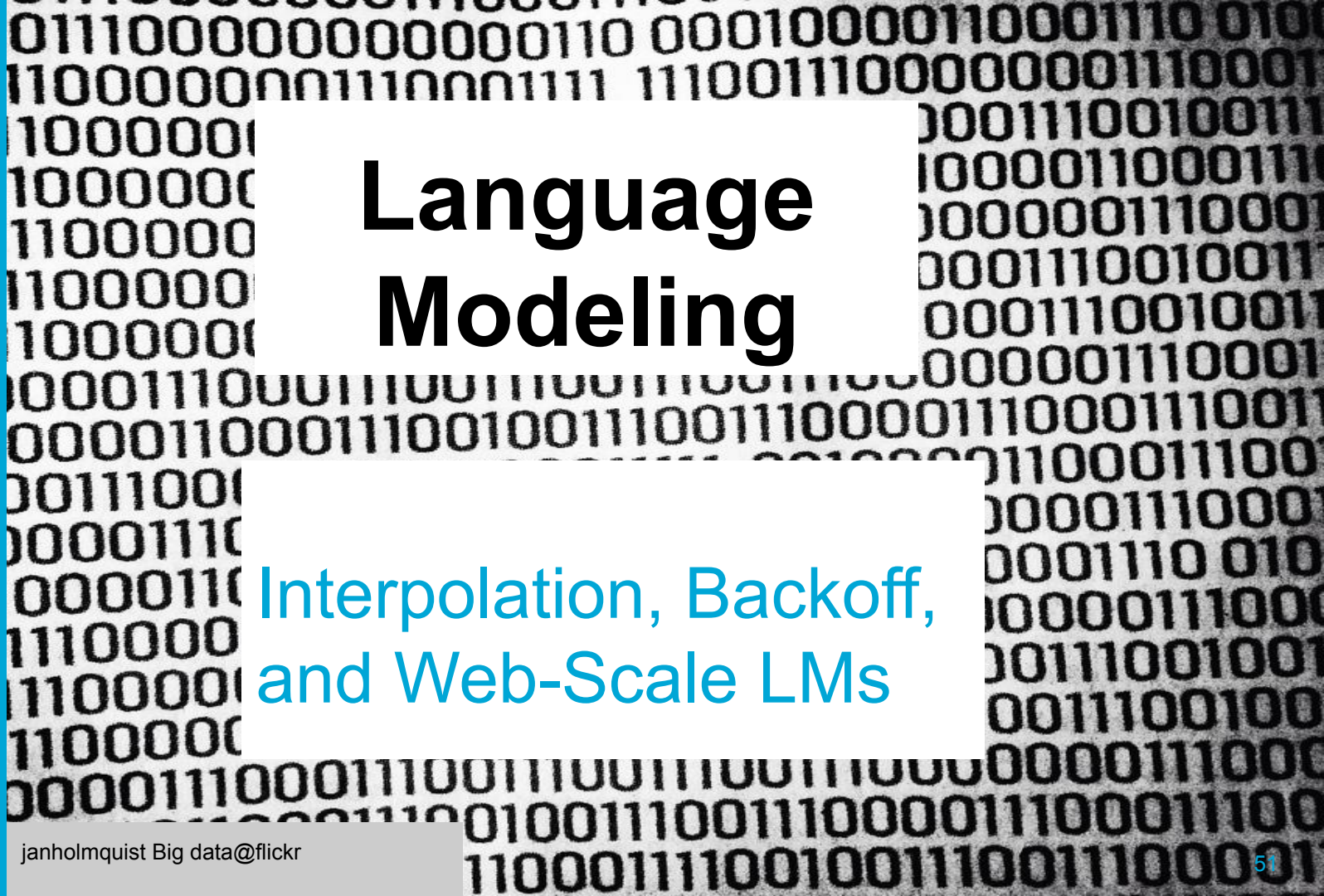
Compare with raw bigram counts

	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

Language Modeling

Interpolation, Backoff, and Web-Scale LMs



Backoff and Interpolation

- Sometimes it helps to use **less** context
  - Condition on less context for contexts you haven't learned much about
- **Backoff:**
  - use trigram if you have good evidence,
  - otherwise bigram, otherwise unigram
- **Interpolation:**
  - mix unigram, bigram, trigram

- Interpolation works better

Linear Interpolation

- Simple interpolation

$$\hat{P}(w_n|w_{n-1}w_{n-2}) = \lambda_1 P(w_n|w_{n-1}w_{n-2}) + \lambda_2 P(w_n|w_{n-1}) + \lambda_3 P(w_n)$$

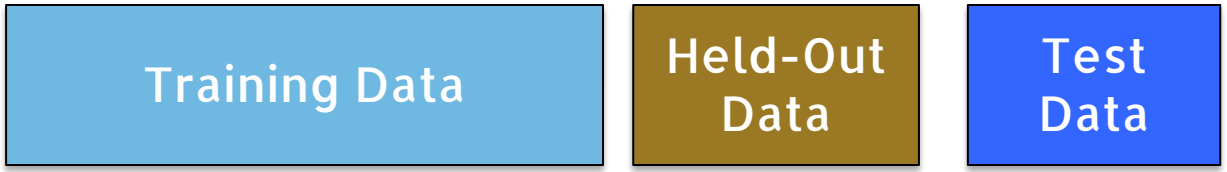
$$\sum_i \lambda_i = 1$$

- Lambdas conditional on context :
  - counts for a particular bigram

$$\hat{P}(w_n|w_{n-2}w_{n-1}) = \lambda_1(w_{n-2}^{n-1})P(w_n|w_{n-2}w_{n-1}) + \lambda_2(w_{n-2}^{n-1})P(w_n|w_{n-1}) + \lambda_3(w_{n-2}^{n-1})P(w_n)$$

How to set the lambdas?

- Use a **held-out** corpus



- Choose λs to maximize the probability of held-out data:
  - Fix the N-gram probabilities (on the training data)
  - Then search for λs that give largest probability to held-out set:

$$\log P(w_1 \ldots w_n \mid M(\lambda_1 \ldots \lambda_k)) = \sum_i \log P_{M(\lambda_1 \ldots \lambda_k)}(w_i \mid w_{i-1})$$

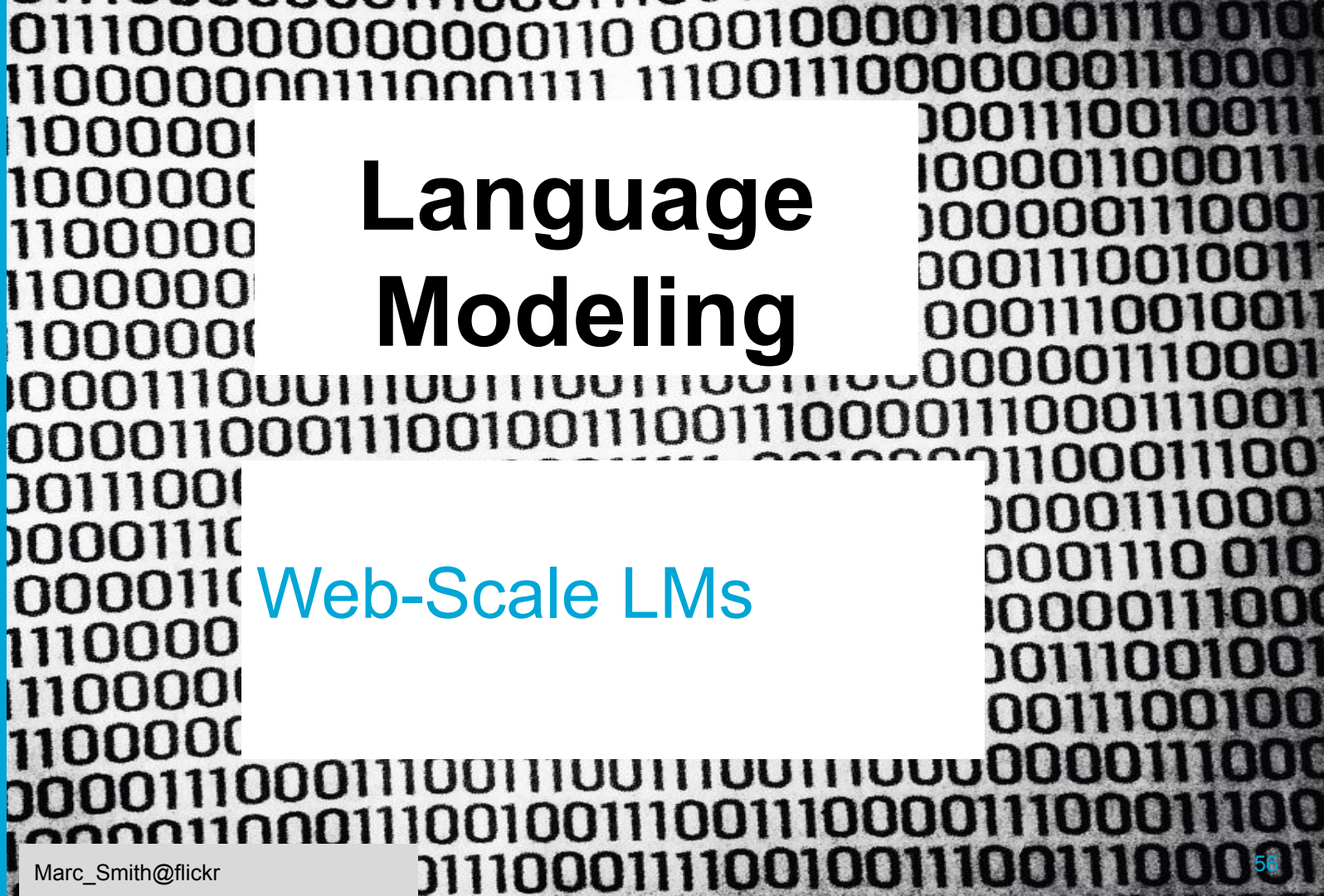
Unknown words: Open versus closed vocabulary tasks

- If we know all the words in advanced
  - Vocabulary V is fixed; Closed vocabulary task
- Often we don't know this
  - **Out Of Vocabulary** = OOV words; Open vocabulary task

- Instead: create an unknown word token <UNK>
  - **Training** of <UNK> probabilities
    - Create a fixed lexicon L of size V
    - At text normalization phase, any training word not in L changed to <UNK>
    - Now we train its probabilities like a normal word
  - At **decoding** time
    - If text input: Use UNK probabilities for any word not in training

Language Modeling

Web-Scale LMs



Marc_Smith@flickr

Huge web-scale n-grams

Pruning

- Only store N-grams with count > threshold.
  - Remove singletons of higher-order n-grams
- Entropy-based pruning
  - select N-grams for pruning such that gap between two models is minimized

$$D(p||p') = -\sum_{w_i,h_j} p(w_i,h_j)[\log p'(w_i|h_j) - \log p(w_i|h_j)]$$

Huge web-scale n-grams

Efficiency

- Efficient data structures like tries (prefix tree)
- Bloom filters: hashing for approximate language models
  - false positives are possible, but never false negatives. If its says it's not there, it's not there.
- Store words as indexes, not strings
  - Use Huffman coding (based on word frequency) to fit large numbers of words into two bytes
- Quantize probabilities (4-8 bits instead of 8-byte float)

Smoothing for Web-scale N-grams

- "Stupid backoff" (Brants *et al*. 2007)
- Use relative frequencies
- First look for the word at the n-gram level
  - if there is no n-gram of that size it will recurse to the (n-1)-gram and multiply its score with 0.4.
  - The recursion stops at unigrams.

$$S(w_i \mid w_{i-k+1}^{i-1}) = \begin{cases} \dfrac{\text{count}(w_{i-k+1}^{i})}{\text{count}(w_{i-k+1}^{i-1})} & \text{if } \text{count}(w_{i-k+1}^{i}) > 0 \\ 0.4 S(w_i \mid w_{i-k+2}^{i-1}) & \text{otherwise} \end{cases}$$

$$S(w_i) = \frac{\text{count}(w_i)}{N}$$

Syntax

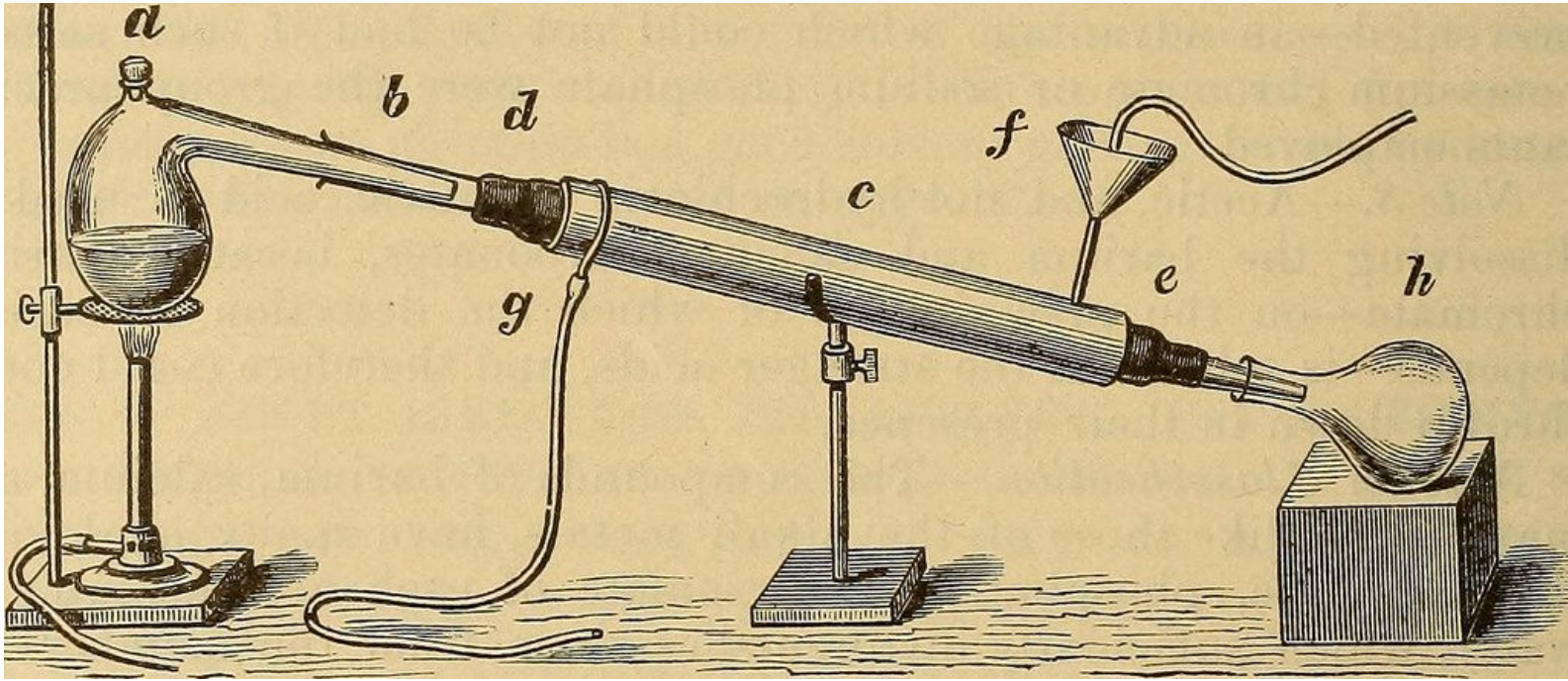
Part 1:

- Part-of-speech (POS)
- Parsing
- Language modeling

Part 2:

- Sentiment analysis
- Named-entity recognition

Applications



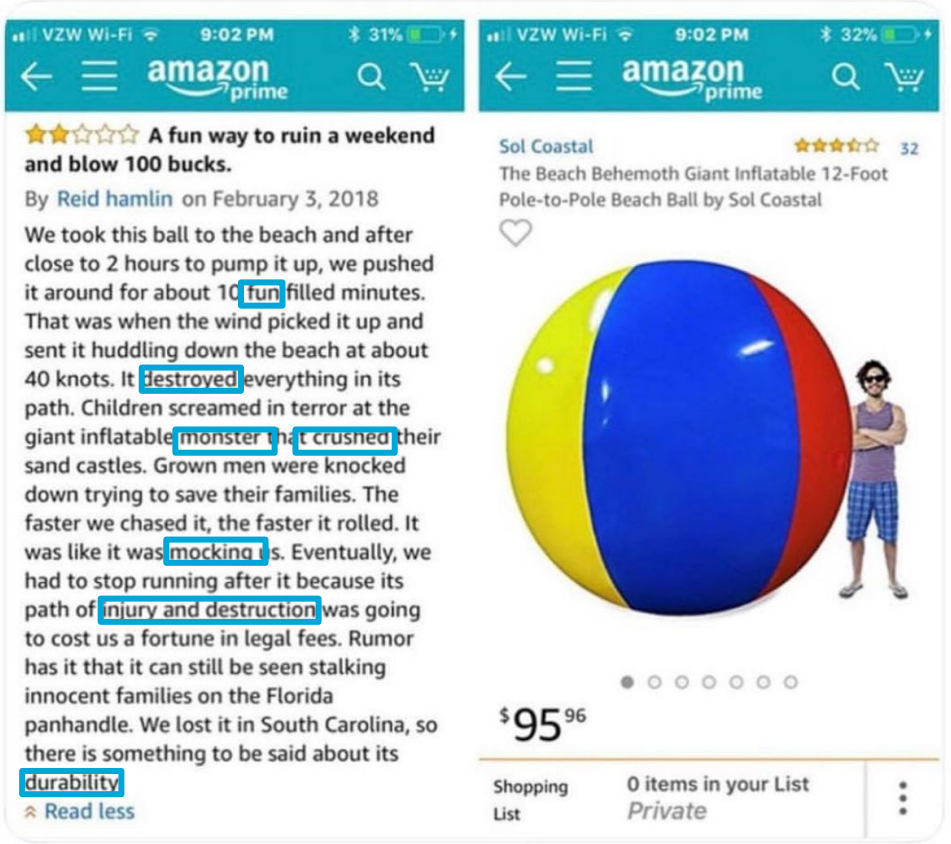
Sentiment analysis

Sentiment analysis

Lexicon based approaches

- Fun, good (+1)
- Terrible, destroy (-1)

Not, never are sentiment shifters (or valence shifters)



Sentiment analysis

Syntactic pattern as a sequence of POS tags (Turney, 2002)

Penn treebank tags: JJ= adjective, NN= noun, VB=verb, RB=adverb
https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

Rule	First word	Second word	Third word (not extracted)
1	JJ	NN or NNS	anything
2	RB, RBR, or RBS	JJ	Not NN nor NNS
3	JJ	JJ	Not NN nor NNS
4	NN or NNS	JJ	Not NN nor NNS
5	RB, RBR, or RBS	VB, VBD, VBN, or VBG	anything

Sentiment analysis

1) Syntactic pattern as a sequence of POS tags (Turney, 2002)

"This piano produces beautiful sounds" ← 1) JJ+NN/NNS

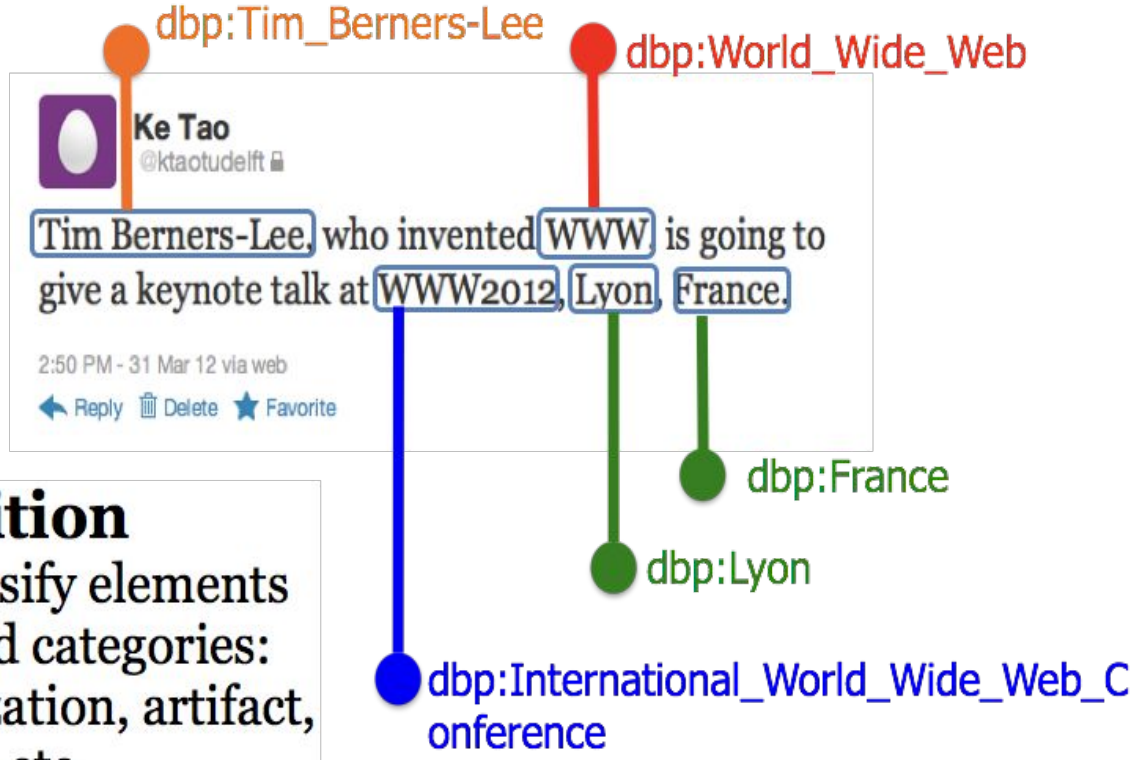
2) Sentiment orientation (SO) using Pointwise Mutual Information Measure (PMI)

$$\operatorname{pmi}(x;y) \equiv \log \frac{p(x,y)}{p(x)p(y)} = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)}.$$

SO(phrase)= PMI(phrase, "excellent") - PMI(phrase, "poor")

3) Average SO score for all phrases gives sentiment to e.g., a review.

Named entity recognition



NER Definition

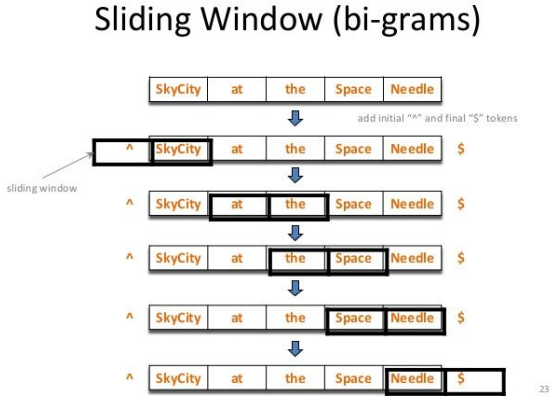
Locate and classify elements into pre-defined categories: person, organization, artifact, event, location, etc.

Named entity recognition

- Detect and classify all **proper names** in a text.
- **Application specific:** commercial products, names of genes, works of art
- **Generic:** names of people, places, and organizations mentioned in news

NER as sequence labeling

- Task similar to POS tagging and syntactic chunking
- Classifier trained to label tokens with tags.
- Features to predict
  - NER type (label)
  - Boundary



NER as sequence labeling

- **Shape** feature
  - Lower case, upper case, expressions that make use of numbers (A9), punctuation (Yahoo!), and typical case alternations (eBay)
- **Gazetter** or **name list**
  - Extensive lists
  - Place names = gazetters, millions of entries
  - Difficult to create and maintain
    - Usefulness varies considerably
    - Gazetters (places) mostly effective
    - Name lists for persons and organizations not nearly as beneficial!

NER as sequence labeling

Training features:

- Shape feature
- Gazetter or name list

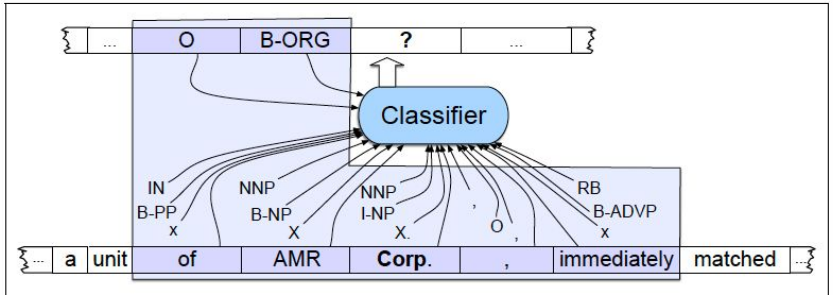
- **Lexical items**
  - Stemmed lexical items
- **Character affixes**: of the target and surrounding words
- **Part of speech of the word**
- **Syntactic chunk labels**: base-phrase chunk label
- **BoW/Bag of n-grams:** words or n-grams occurring in the surrounding context

Techniques

$P(c,d)$

$P(c \mid d)$

- **Hidden-markov models (HMM)**
  - Generative model: **joint** probability distribution
  - observation likelihood and the prior
  - Uses Bayes rule to model most probably PoS tag sequence
- **Maximal entropy markov model (MEMM)**
  - Discriminative model: **conditional** probability
  - *"the probability of state given observation"*.
  - Compute the posterior of each state, conditioned on the previous state and current observation



Named entity recognition

- **Multi-way classification task**

In 1917, Einstein applied the general theory of relativity to model the large-scale structure of the universe. He was visiting the United States when Adolf Hitler came to power in 1933 and did not go back to Germany, where he had been a professor at the Berlin Academy of Sciences. He settled in the U.S., becoming an American citizen in 1940. On the eve of World War II, he endorsed a letter to President Franklin D. Roosevelt alerting him to the potential development of "extremely powerful bombs of a new type" and recommending that the U.S. begin similar research. This eventually led to what would become the Manhattan Project. Einstein supported defending the Allied forces, but largely denounced using the new discovery of nuclear fission as a weapon. Later, with the British philosopher Bertrand Russell, Einstein signed the Russell-Einstein Manifesto, which highlighted the danger of nuclear weapons. Einstein was affiliated with the Institute for Advanced Study in Princeton, New Jersey, until his death in 1955.

Tag colours:
LOCATION TIME PERSON ORGANIZATION MONEY PERCENT DATE

Summary

- Part-of-speech (POS) tagging
- Language modeling

Applications in...

- Sentiment analysis
- Named-entity recognition

Next deadlines:

- Review P1: handed out Feb 14, due February 21.

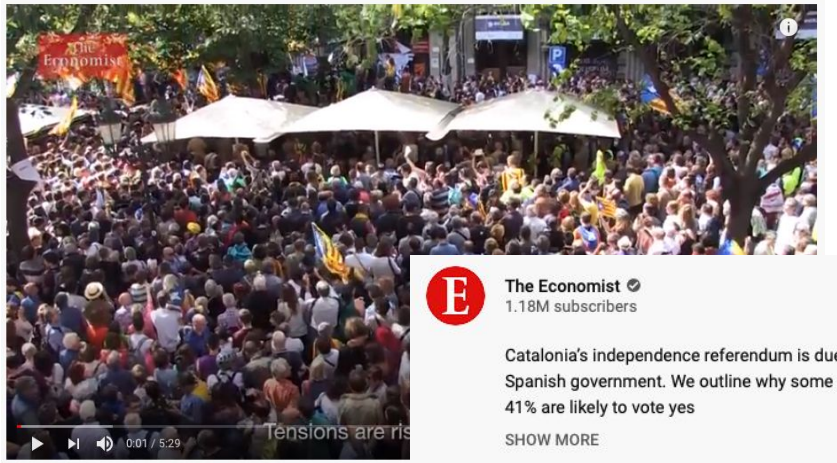
  Fokkens, Antske, et al. "Offspring from reproduction problems: What replication failure teaches us." Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vol. 1. 2013
- Decide on a group: due Friday **Feb. 14th**
- NLP project proposal: **due February 21**

TU Delft

Questions?

Possible project: Emotion/sentiment analysis

Identify emotions and/or sentiments in video comments
Controversial topics



The Economist
1.18M subscribers
SUBSCRIBE

Catalonia's independence referendum is due to take place on October 1st, in clear defiance of the Spanish government. We outline why some Catalans want independence from Spain, and why only 41% are likely to vote yes

SHOW MORE

1,528 Comments SORT BY

Possible project: Fake News Challenge

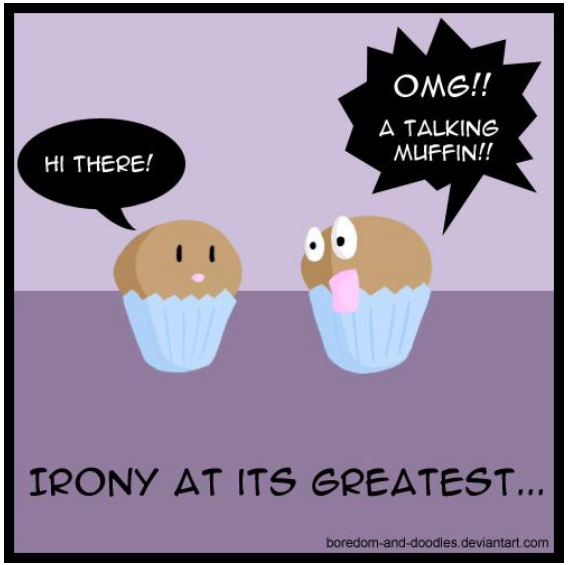
http://www.fakenewschallenge.org

https://github.com/FakeNewsChallenge

Possible project: Irony Detection on English tweets

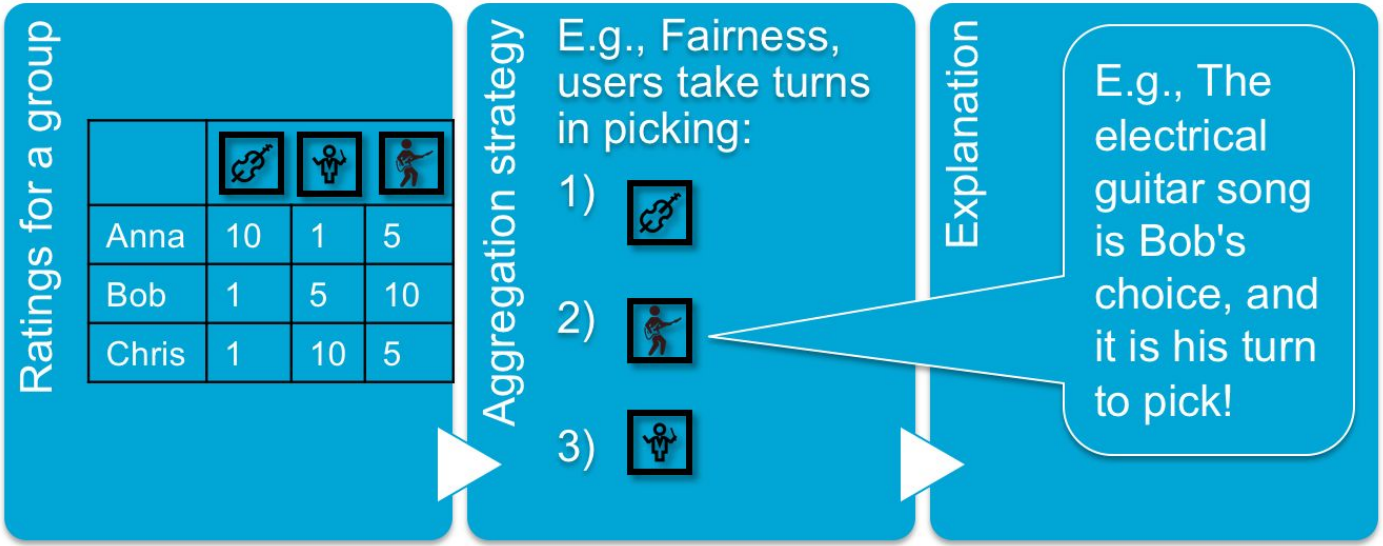
https://competitions.codalab.org/competitions/17468

https://github.com/Cyvhee/SemEval2018-Task3/



Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. Semeval-2018 Task 3: Irony detection in English Tweets. In *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA, June 2018.

Possible project:
Explaining Recommendations



Next week...

- **Semantics**
  - Word sense disambiguation
  - Other features for sentiment analysis

TU Delft