Amirkabir University of Technology
(Tehran Polytechnic)

Natural Language Processing

Lecture 4: Probabilistic Language Model

Amirkabir University of Technology
Dr Momtazi

Outline

- **Motivation**
- Estimation
- Smoothing

Language Modeling

- Finding the probability of a sentence or a sequence of words

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

- Applications:
  - Word prediction
  - Speech recognition
  - Machine translation
  - Spell checker

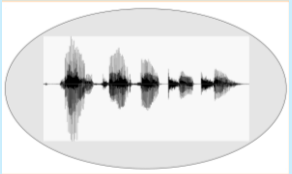
Applications

- Word Prediction

“natural language” ⇒ “processing”
“management”

Applications

- Speech recognition



⇒

"Computers can recognize speech."
"Computers can wreck a nice peach."

Applications

- Machine translation

“The cat eats ...”	$\Rightarrow$	*“Die Katze frisst ...”* *“Die Katze isst ...”*

Applications

- Spell checker

"I want to adver this project." $\Rightarrow$ *"advert"*
"adverb"

Outline

- Motivation
- **Estimation**
- Smoothing

Corpus

- Probabilities are based on counting things
- Counting of thing in natural language is based on a corpus (plural: corpora)
- A computer-readable collection of text or speech
  - The Brown Corpus
    - A million-word collection of samples
    - 500 written texts from different genres
      (newspaper, fiction, non-fiction, academic, ...)
    - Assembled at Brown University in 1963-1964
  - The Switchboard Corpus
    - A collection of 240 hours of telephony conversations
    - 3 million words in 2430 conversations averaging 6 minutes each
    - Collected in early 1990s

Text Corpora I

- American National Corpus
- Bank of English
- British National Corpus
- Bergen Corpus of London Teenage Language (COLT)
- Brown Corpus, forming part of the "Brown Family" of corpora, together with LOB, Frown and F-LOB
- Corpus of Contemporary American English (COCA) 425 million words, 1990–2011. Freely searchable online
- Corpus Resource Database (CoRD), more than 80 English language corpora.

Text Corpora II

- Coruña Corpus, a corpus of late Modern English scientific writing covering the period 1700-1900, developed by the Muste research group at the University of A Coruña
- GUM corpus, the open source Georgetown University Multilayer corpus, with very many annotation layers
- Google Books Ngram Corpus
- International Corpus of English
- Oxford English Corpus
- RE3D (Relationship and Entity Extraction Evaluation Dataset)
- Santa Barbara Corpus of Spoken American English
- Scottish Corpus of Texts & Speech

Text Corpora III

- Apache Software Foundation Public Mail Archives: all publicly available Apache Software Foundation mail archives as of July 11, 2011 (200 GB)
- Blog Authorship Corpus: consists of the collected posts of 19,320 bloggers gathered from blogger.com in August 2004. 681,288 posts and over 140 million words. (298 MB)
- Amazon Fine Food Reviews [Kaggle]: consists of 568,454 food reviews Amazon users left up to October 2012. Paper. (240 MB)
- Amazon Reviews: Stanford collection of 35 million amazon reviews. (11 GB)
- ArXiv: All the Papers on archive as fulltext (270 GB) + sourcefiles (190 GB).

Text Corpora IV

- CLiPS Stylometry Investigation (CSI) Corpus: a yearly expanded corpus of student texts in two genres: essays and reviews. The purpose of this corpus lies primarily in stylometric research, but other applications are possible. (on request)
- ClueWeb09 FACC: ClueWeb09 with Freebase annotations (72 GB)
- ClueWeb11 FACC: ClueWeb11 with Freebase annotations (92 GB)
- Common Crawl Corpus: web crawl data composed of over 5 billion web pages (541 TB)
- Cornell Movie Dialog Corpus: contains a large metadata-rich collection of fictional conversations extracted from raw movie scripts: 220,579 conversational exchanges between 10,292 pairs of movie characters, 617 movies (9.5 MB)

Word Occurrence

- A language consist of a set of V words (Vocabulary)
- A text is a sequence of the words from the vocabulary

- A word can occur several times in a text
  - Word Token: each occurrence of words in text
  - Word Type: each unique occurrence of words in the text

Word Occurrence

Example:
This is a sample text from a book that is read every day

Word Occurrence

Example:
This is a sample text from a book that is read every day

Word Tokens: 13

Word Types: 11

Counting

- Brown
  - 1,015,945 word tokens
  - 47,218 word types

- Google N-gram
  - 1,024,908,267,229 word tokens
  - 13,588,391 word types

Counting

- Brown
  - 1,015,945 word tokens
  - 47,218 word types

- Google N-gram
  - 1,024,908,267,229 word tokens
  - 13,588,391 word types

That seems like a lot of types...

Even large dictionaries of English have only around 500k types.

Why so many here?

Numbers

Misspellings

Names

Acronyms

Bayes Decomposition

- Write joint probability as product of conditional probabilities

$$P(w_1, w_2) = P(w_1) \cdot P(w_2 | \ w_1)$$

$$P(w_1,w_2,w_3,w_4) = P(w_1) \cdot P(w_2 \mid w_1) \cdot P(w_3 \mid w_1,w_2) \cdot P(w_4 \mid w_1,w_2,w_3)$$

$$P(w_1,w_2,\ldots, w_n) = P(w_1) \cdot P(w_2 \mid w_1) \cdot P(w_3 \mid w_1,w_2) \cdots P(w_n \mid w_1,w_2,\ldots, w_{n\text{-}1})$$

$$P(S) = (w_1) \cdot P(w_2 \mid w_1) \cdot P(w_3 \mid w_1,w_2) \cdots P(w_n \mid w_1,w_2,\ldots, w_{n\text{-}1})$$

$$P(S) = \prod_{i=1}^{n} P(w_i \mid w_1,w_2,\ldots, w_{i\text{-}1})$$

Conditional Probability

$$P(S) = \prod_{i=1}^{n} P(w_i \mid w_1, w_2, \ldots, w_{i-1})$$

P(Computer, can, recognize, speech) =

P(Computer)·

P(can|Computer)·

P(recognize|Computer,can)·

P(speech|Computer,can,recognize)

Maximum Likelihood Estimation

P(speech|Computer can recognize)

$$\text{P(speech|Computer can recognize)} = \frac{\#(\text{Computer can recognize speech})}{\#(\text{Computer can recognize})}$$

- Too many phrases
- Limited text for estimating the probability

  => Making a simplification assumption

Markov Assumption

$$P(S) = \prod_{i=1}^{n} P(w_i | w_1, w_2, \ldots, w_{i-1})$$

$$P(S) = \prod_{i=1}^{n} P(w_i | w_{i-1})$$

$P(Computer; can; recognize; speech) =$
$P(Computer) \cdot P(can|Computer) \cdot P(recognize|can) \cdot P(speech|recognize)$

$$(speech|recognize) = \frac{\#(\text{recognize speech})}{\#(\text{recognize})}$$

N-gram Model

- Unigram $P(S) = \prod_{i=1}^{n} P(w_i)$
- Bigram $P(S) = \prod_{i=1}^{n} P(w_i \mid w_{i-1})$
- Trigram $P(S) = \prod_{i=1}^{n} P(w_i \mid w_{i-2} w_{i-1})$

- N-gram $P(S) = \prod_{i=1}^{n} P(w_i \mid w_1 w_2, \ldots, w_{i-1})$

Maximum Likelihood

<s> I saw the boy </s>

<s> the man is working </s>

<s> I walked in the street </s>

Vocab:

I saw the boy man is working walked in street

boy I in is man saw street the walked working

Maximum Likelihood

<s> I saw the boy </s>

<s> the man is working </s>

<s> I walked in the street </s>

boy	I	In	Is	Man	saw	Street	The	Walke d	workin g
1	2	1	1	1	1	1	3	1	1

Maximum Likelihood

Boy	I	In	Is	Man	saw	Street	The	Walked	working
1	2	1	1	1	1	1	3	1	1

	Boy	I	In	Is	Man	saw	Street	The	Walked	working
Boy										
I						1			1	
in								1		
is										1
man				1						
saw								1		
street										
the	1				1		1			
walked			1							
working										

Maximum Likelihood

<s> I saw the man </s>

P(S) = P(I) · P(saw|I) · P(the|saw) · P(man|the)

$$P(S) = \frac{\#(I)}{\#} \cdot \frac{\#(I\ saw)}{\#(I)} \cdot \frac{\#(saw\ the)}{\#(saw)} \cdot \frac{\#(\ the\ man)}{\#(the)}$$

$$P(S) = \frac{2}{13} \cdot \frac{1}{2} \cdot \frac{1}{1} \cdot \frac{1}{3}$$

Outline

- Motivation
- Estimation
- **Smoothing**

Maximum Likelihood

<s> I saw the man </s>

P(S) = P(I) · P(saw|I) · P(the|saw) · P(man|the)

$$P(S) = \frac{\#(I)}{\#} \cdot \frac{\#(I\ saw)}{\#(I)} \cdot \frac{\#(saw\ the)}{\#(saw)} \cdot \frac{\#(\ the\ man)}{\#(the)}$$

$$P(S) = \frac{2}{13} \cdot \frac{1}{2} \cdot \frac{1}{1} \cdot \frac{1}{3}$$

Zero Probability

<s> I saw the man in the street </s>

boy	I	in	is	man	saw	street	the	walked	working
1	2	1	1	1	1	1	3	1	1

	boy	I	in	is	man	saw	street	the	walked	working
boy	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	1	0	0	1	0
in	0	0	0	0	0	0	0	1	0	0
is	0	0	0	0	0	0	0	0	0	1
man	0	0	0	1	0	0	0	0	0	0
saw	0	0	0	0	0	0	0	1	0	0
street	0	0	0	0	0	0	0	0	0	0
the	1	0	0	0	1	0	1	0	0	0
walked	0	0	1	0	0	0	0	0	0	0
working	0	0	0	0	0	0	0	0	0	0

Maximum Likelihood

<s> I saw the man in the street </s>

P(S) = P(I)· P(saw|I)· P(the|saw)· P(man|the)· P(in|man) · P(the|in) · P(street|the)

$$P(S) = \frac{\#(I)}{\#} \cdot \frac{\#(I\ saw)}{\#(I)} \cdot \frac{\#(saw\ the)}{\#(saw)} \cdot \frac{\#(the\ man)}{\#(the)} \cdot \frac{\#(man\ in)}{\#(\mathrm{man})} \cdot \frac{\#(in\ the)}{\#(\mathrm{in})} \cdot \frac{\#(the\ street)}{\#(the)}$$

$$P(S) = \frac{2}{13} \cdot \frac{1}{2} \cdot \frac{1}{1} \cdot \frac{1}{3} \cdot \frac{0}{1} \cdot \frac{1}{1} \cdot \frac{1}{3}$$

Smoothing

- Giving a small probability to all as unseen n-grams

Laplace Smoothing

- Add one to all counts (Add-one)

Laplace Smoothing

- Add one to all counts (Add-one)

	boy	I	in	is	man	saw	street	the	walked	working
boy	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	1	0	0	1	0
in	0	0	0	0	0	0	0	1	0	0
is	0	0	0	0	0	0	0	0	0	1
man	0	0	0	1	0	0	0	0	0	0
saw	0	0	0	0	0	0	0	1	0	0
street	0	0	0	0	0	0	0	0	0	0
the	1	0	0	0	1	0	1	0	0	0
walked	0	0	1	0	0	0	0	0	0	0
working	0	0	0	0	0	0	0	0	0	0

Laplace Smoothing

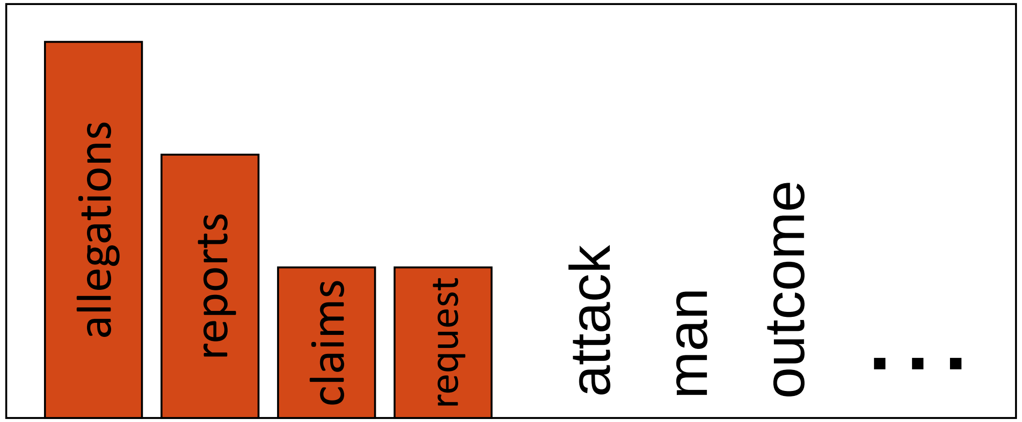
	boy	I	in	is	man	saw	street	the	walked	working
boy	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	1	0	0	1	0
in	0	0	0	0	0	0	0	1	0	0
is	0	0	0	0	0	0	0	0	0	1
man	0	0	0	1	0	0	0	0	0	0
saw	0	0	0	0	0	0	0	1	0	0
street	0	0	0	0	0	0	0	0	0	0
the	1	0	0	0	1	0	1	0	0	0
walked	0	0	1	0	0	0	0	0	0	0
working	0	0	0	0	0	0	0	0	0	0

	boy	I	in	is	man	saw	street	the	walked	working
boy	1	1	1	1	1	1	1	1	1	1
I	1	1	1	1	1	2	1	1	2	1
in	1	1	1	1	1	1	1	2	1	1
is	1	1	1	1	1	1	1	1	1	2
man	1	1	1	2	1	1	1	1	1	1
saw	1	1	1	1	1	1	1	2	1	1
street	1	1	1	1	1	1	1	1	1	1
the	2	1	1	1	2	1	2	1	1	1
walked	1	1	2	1	1	1	1	1	1	1
working	1	1	1	1	1	1	1	1	1	1

The intuition of smoothing

- When we have sparse statistics:

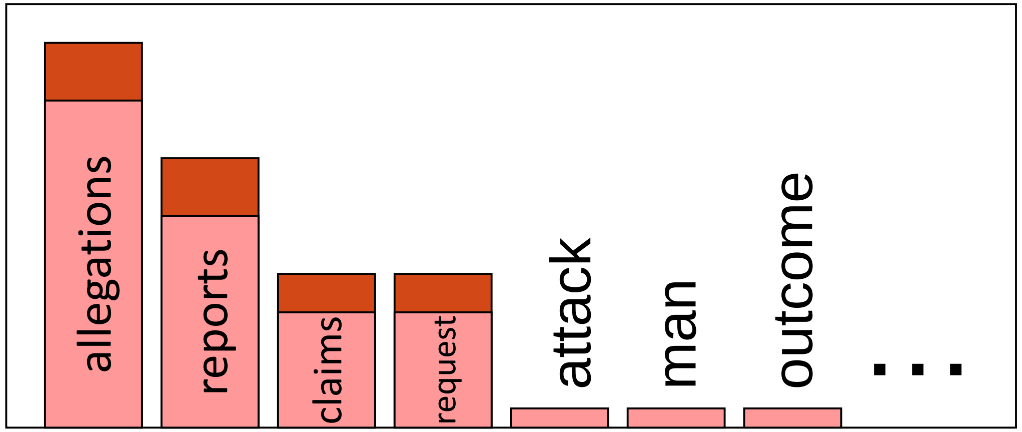
  P(w | denied the)
  3 allegations
  2 reports
  1 claims
  1 request
  ---
  7 total



- 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



Laplace Smoothing

- Add one to all counts (Add-one)
- $P(w_i|w_{i-1}) = \frac{\#(w_{i-1}, w_i)}{\#(w_{i-1})} \Longrightarrow P(w_i|w_{i-1}) = \frac{\#(w_{i-1}, w_i)+1}{\#(w_{i-1})+V}$

Smoothing

- Interpolation and Back-off Smoothing
  - Use a background probability

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

Back-off

$$P(w_i|w_{i-1}) = \begin{cases} \frac{\#(w_{i-1}, w_i)}{\#(w_{i-1})} & \text{if } \#(w_{i-1}, w_i) > 0 \\ P_{BG} & \text{Otherwise} \end{cases}$$

Smoothing

- Interpolation and Back-off Smoothing
  - Use a background probability

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

Interpolation

$$P(w_i|w_{i-1}) = \lambda_1 \frac{\#(w_{i-1},w_i)}{\#(w_{i-1})} + \lambda_2 P_{BG} \qquad \sum \lambda = 1$$

Parameter Tuning

Background Probability

Background Probability

- Lower levels of n-gram can be used as background probability
- trigram → bigram
- bigram → unigram
- unigram → zerogram (1/V)

Background Probability

- Lower levels of n-gram can be used as background probability
- trigram → bigram
- bigram → unigram
- unigram → zerogram (1/V)

Back-off

$$P(w_i|w_{i-1}) = \begin{cases} \dfrac{\#(w_{i-1},w_i)}{\#(w_{i-1})} & \text{if } \#(w_{i-1},w_i) > 0 \\ P_{BG} & \text{Otherwise} \end{cases}$$

Background Probability

Back-off

$$P(w_i|w_{i-1}) = \begin{cases} \frac{\#(w_{i-1},w_i)}{\#(w_{i-1})} & \text{if } \#(w_{i-1}, w_i)>0 \\ P_{BG} & \text{Otherwise} \end{cases}$$

$$P(w_i) = \begin{cases} \frac{\#(w_i)}{N} & \text{if } \#(w_i)>0 \\ 1/V & \text{Otherwise} \end{cases}$$

Background Probability

Interpolation

$$P(w_i|w_{i-1}) = \lambda_1 \frac{\#(w_{i-1}, w_i)}{\#(w_{i-1})} + \lambda_2 P_{BG}$$

$$P(\mathrm{w_i}) = \lambda_1 \frac{\#(w_i)}{N} + \lambda_2 . (1/V)$$

Advanced Smoothing

- Bayesian Smoothing with Dirichlet Prior
- Absolute Discounting
- Kneser-Ney Smoothing
- Bayesian Smoothing based on Pitman-Yor Processes

Bayesian Smoothing with Dirichlet Prior

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

$$P(w_i|w_{i-1}) = \frac{\#(w_{i-1},w_i)+k}{\#(w_{i-1})+kV}$$

$$P(w_i|w_{i-1}) = \frac{\#(w_{i-1},w_i)+\mu\left(\frac{1}{v}\right)}{\#(w_{i-1})+\mu}$$

μ=kv

$$P(w_i|w_{i-1}) = \frac{\#(w_{i-1},w_i)+\mu\, P_{BG}}{\#(w_{i-1})+\mu}$$

Absolute Discounting

$$P(w_i|w_{i-1}) = \begin{cases} \dfrac{\#(w_{i-1},w_i)}{\#(w_{i-1})} & \text{if } \#(w_{i-1},w_i)>0 \\ P_{BG} & \text{Otherwise} \end{cases}$$

$$P(w_i|w_{i-1}) = \begin{cases} \dfrac{\#(w_{i-1},w_i)-\delta}{\#(w_{i-1})} & \text{if } \#(w_{i-1},w_i)>0 \\ \alpha P_{BG} & \text{Otherwise} \end{cases}$$

Absolute Discounting

$$P(w_i|w_{i-1}) = \frac{\#(w_{i-1}, w_i) - \delta}{\#(w_{i-1})} + \alpha P_{BG}$$

$$\alpha = \frac{\delta}{\#(w_{i-1})} \cdot B$$

B : the number of times $\#(w_i, w_{i-1}) > 0$
(the number of times that we applied discounting)

$$P(w_i|w_{i-1}) = \frac{\max(\#(w_{i-1}, w_i) - \delta, 0)}{\#(w_{i-1})} + \alpha P_{BG}$$

Kneser-Ney Smoothing

- Estimation base on the lower-order n-gram

" I cannot see without my reading ... "	$\Rightarrow$	*"Francisco"* *"glasses"*

- Observations:
  - "Francisco" is more common than "glasses"
  - But "Francisco" always follows "San"
  - "Francisco" is not a novel continuation for a text
- Solution:
  - Instead of $P(w)$: "How likely is w to appear in a text"
  - $P_{continuation}(w)$: "How likely is w to appear as a novel continuation"
  - Count the number of words types that w appears after them

$$P_{continuation}(w) \propto |w_{i-1}: \#(w_{i-1}; w_i) > 0|$$

Kneser-Ney Smoothing

- How many times does w appear as a novel continuation

$$P_{continuation}(w) \propto |w_{i-1} : \#(w_{i-1}, w_i) > 0|$$

- Normalized by the total number of bigram types

$$P_{continuation}(w) = \frac{|w_{i-1} : \#(w_{i-1}, w_i) > 0|}{|(w_{j-1}, w_j) : \#(w_{j-1}, w_j) > 0|}$$

- Alternatively: normalized by the number of words preceding all words

$$P_{continuation}(w) = \frac{|w_{i-1} : \#(w_{i-1}, w_i) > 0|}{\Sigma_{w'} |w'_{i-1} : \#(w'_{i-1}, w'_i) > 0|}$$

Kneser-Ney Smoothing

$$P(w_i|w_{i-1}) = \frac{\max(\#(w_{i-1}, w_i) - \delta, 0)}{\#(w_{i-1})} + \alpha P_{BG}$$

$$P(w_i|w_{i-1}) = \frac{\max(\#(w_{i-1}, w_i) - \delta, 0)}{\#(w_{i-1})} + \alpha P_{continuation}$$

$$\alpha = \frac{\delta}{\#(w_{i-1})} \cdot B$$

B : the number of times $\#(w_{i-1}, w_i) > 0$

Bayesian Smoothing based on Pitman-Yor Processes

- Improving the Dirichlet prior by using a discounting parameter deriving from absolute discounting method
- Dirichlet prior

$$P(w_i|w_{i-1}) = \frac{\#(w_{i-1},w_i)+\mu\, P_{BG}}{\#(w_{i-1})+\mu}$$

- Absolute discounting

$$P(w_i|w_{i-1}) = \frac{\#(w_{i-1},w_i) - \delta + (\delta.B)P_{BG}}{\#(w_{i-1})}$$

- Combined

$$P(w_i|w_{i-1}) = \frac{\#(w_{i-1},w_i) - \delta + (\mu + \delta.B)P_{BG}}{\#(w_{i-1}) + \mu}$$

Bayesian Smoothing based on Pitman-Yor Processes

$$P(w_i|w_{i-1}) = \frac{\#(w_{i-1}, w_i) - \ \delta t + (\mu + \delta t.)P_{BG}}{\#(w_{i-1}) + \mu}$$

- t: discounting weight
- t.: total amount of applied discounting

Bayesian Smoothing based on Pitman-Yor Processes

- Using different discounting value for each word based on the frequency of that word

$$P(w_i|w_{i-1}) = \frac{\#(w_{i-1}, w_i) - \delta t + (\mu + \delta t.)P_{BG}}{\#(w_{i-1}) + \mu}$$

- t: discounting weight
- t.: total amount of applied discounting
- t = 1 → basic combined model
- μ = 0 → absolute discounting method

Bayesian Smoothing based on Pitman-Yor Processes

- Calculating parameter t is the most important and computationally expensive part of the formula
- Idea for a near optimum estimation of t: Generating a power-law distribution in the language model, which is one of the statistical properties of word frequencies in natural language

$$T=0 \qquad \text{if } \#(w_{i-1}, w_i) = 0$$

$$t = f(\#(w_{i-1}, w_i)) = \left(\#(w_{i-1}, w_i)\right)^{\delta} \qquad \text{if } \#(w_{i-1}, w_i) > 0$$

Further Reading

- Speech and Language Processing (3rd ed. draft)
  - Chapter 3