Statistical Natural Language Processing

Lecture 4: Language Modeling

Dr. Momtazi
Amirkabir University of Technology

Outline

1. Motivation

2. Estimation

3. Smoothing

Outline

1. Motivation

2. Estimation

3. Smoothing

Language Modeling

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

$$P(S) = P(w_1, w_2, w_3, ..., w_n)$$

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

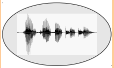
Applications

- Word Prediction

"natural language ..." $\Rightarrow$ *"processing"*
"management"

Applications

- Speech recognition

 $\Rightarrow$ *"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 <u>adver</u> this project." $\Rightarrow$ *"advert"*
"adverb"

Outline

1 Motivation

2 Estimation

3 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

Corpus

- Text Corpora
  - The Brown Corpus
  - Corpus of Contemporary American English
  - The British National Corpus
  - The International Corpus of English
  - The Google *N*-gram Corpus

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 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

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

Language Modeling

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

$$P(S) = P(w_1, w_2, w_3, ..., w_n)$$

P(Computer, can, recognize, speech)

Bayes Decomposition

- Write joint probability as product of conditional probabilities

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

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

$$P(w_1, w_2, ...w_n) = P(w_1) \cdot P(w_2|w_1) \cdot P(w_3|w_1, w_2) \cdots P(w_n|w_1, w_2, w_3, ..., w_{n-1})$$

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

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

Conditional Probability

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

$P(Computer, can, recognize, speech) =$

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

Maximum Likelihood Estimation

$$P(speech|Computer\ can\ recognize)$$

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

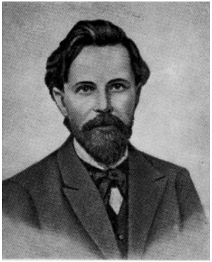
- Too many phrases
- Limited text for estimating the probability

$\Rightarrow$ Making a simplification assumption

Markov Assumption

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

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



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

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

N-gram Model

Unigram $P(S) = \prod_{i=1}^{n} P(w_i)$

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

Trigram $P(S) = \prod_{i=1}^{n} P(w_i|w_{i-2}, w_{i-1})$

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

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	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 </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

$$P(S) = P(I) \cdot P(saw|I) \cdot P(the|saw) \cdot P(man|the)$$

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

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

Outline

1. Motivation
2. Estimation
3. Smoothing

Maximum Likelihood

<s> I saw the man </s>

$$P(S) = P(I) \cdot P(saw|I) \cdot P(the|saw) \cdot P(man|the)$$

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

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

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

$$P(S) = P(I) \cdot P(saw|I) \cdot P(the|saw) \cdot P(man|the) \cdot P(in|man) \cdot P(the|in) \cdot P(street|the)$$

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

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

Smoothing

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

Laplace Smoothing

- Add one to all counts (Add-one)

	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

Laplace Smoothing

- Add one to all counts (Add-one)

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

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

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

$$\sum \lambda = 1$$

Parameter Tuning

Background Probability

Background Probability

- Lower levels of *n*-gram can be used as background probability
  - trigram $\rightarrow$ bigram
  - bigram $\rightarrow$ unigram
  - unigram $\rightarrow$ zerogram ($\frac{1}{V}$)

Back-off

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

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

Background Probability

- Lower levels of *n*-gram can be used as background probability
  - trigram $\rightarrow$ bigram
  - bigram $\rightarrow$ unigram
  - unigram $\rightarrow$ zerogram ($\frac{1}{V}$)

Interpolation

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

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

$$P(w_i|w_{i-1}) = \lambda_1 \frac{\#(w_{i-1}, w_i)}{\#(w_{i-1})} + \lambda_2 \frac{\#(w_i)}{N} + \lambda_3 \frac{1}{V}$$

Advanced Smoothing

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

Bayesian Smoothing with Dirichlet Prior

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

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

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

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

Absolute Discounting

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

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

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 $\#(w_{i-1}, w_i) > 0$

(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

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

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

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

  - Combined

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

Bayesian Smoothing based on Pitman-Yor Processes

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

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

t: discounting weight
$t_{.}$: total amount of applied discounting

$t = 1$ $\rightarrow$ basic combined model

$\mu = 0$ $\rightarrow$ 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

$$\begin{cases} t = 0 & \text{if } \#(w_{i-1}, w_i) = 0 \\ t = f(\#(w_{i-1}, w_i)) = (\#(w_{i-1}, w_i))^{\delta} & \text{if } \#(w_{i-1}, w_i) > 0 \end{cases}$$

Further Reading

- Speech and Language Processing
  - Chapter 4