N-gram Language Models

Pawan Goyal

CSE, IITKGP

Week 2: Lecture 4

Context Sensitive Spelling Correction

The office is about fifteen minuets from my house

Context Sensitive Spelling Correction

The office is about fifteen minuets from my house

min·u·et *noun* \ˌmin-yə-ˈwet\

: a slow, graceful dance that was popular in the 17th and 18th centuries

: the music for a minuet

Context Sensitive Spelling Correction

The office is about fifteen minuets from my house

min·u·et *noun* \ˌmin-yə-ˈwet\

: a slow, graceful dance that was popular in the 17th and 18th centuries

: the music for a minuet

Use a Language Model

P(about fifteen **minutes** from) > P(about fifteen **minuets** from)

Probablilistic Language Models: Applications

Speech Recognition

- P(I saw a van) $>>$ P(eyes awe of an)

Probablilistic Language Models: Applications

Speech Recognition

- P(I saw a van) $>>$ P(eyes awe of an)

Machine Translation

Which sentence is more plausible in the target language?

- P(**high** winds) > P(**large** winds)

Probablilistic Language Models: Applications

Speech Recognition

- P(I saw a van) $>>$ P(eyes awe of an)

Machine Translation

Which sentence is more plausible in the target language?

- P(**high** winds) > P(**large** winds)

Other Applications

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

Completion Prediction

- Language model also supports predicting the completion of a sentence.
  - Please turn off your cell ...
  - Your program does not ...

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.

Probabilistic Language Modeling

- **Goal:** Compute the probability of a sentence or sequence of words:

$$P(W) = P(w_1, w_2, w_3, \ldots, w_n)$$

Probabilistic Language Modeling

- **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:

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

Probabilistic Language Modeling

- **Goal:** Compute the probability of a sentence or sequence of words:

$$P(W) = P(w_1, w_2, w_3, \ldots, w_n)$$

- **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**

Computing $P(W)$

How to compute the joint probability

P(about, fifteen, minutes, from)

Computing $P(W)$

How to compute the joint probability

P(about, fifteen, minutes, from)

Basic Idea

Rely on the Chain Rule of Probability

The Chain Rule

Conditional Probabilities

$$P(B|A) = \frac{P(A,B)}{P(A)}$$

The Chain Rule

Conditional Probabilities

$$P(B|A) = \frac{P(A,B)}{P(A)}$$

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

The Chain Rule

Conditional Probabilities

$$P(B|A) = \frac{P(A,B)}{P(A)}$$

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

More Variables

$$P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)$$

The Chain Rule

Conditional Probabilities

$$P(B|A) = \frac{P(A,B)}{P(A)}$$

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

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

$$P(x_1,x_2,\ldots,x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1,x_2)\ldots P(x_n|x_1,\ldots,x_{n-1})$$

Probability of words in sentences

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

P("about fifteen minutes from") =

Probability of words in sentences

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

P("about fifteen minutes from") =

P(about) x P(fifteen | about) x P(minutes | about fifteen) x P(from | about fifteen minutes)

Estimating These Probability Values

Count and divide

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

Estimating These Probability Values

Count and divide

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

What is the problem

We may never see enough data for estimating these

Markov Assumption

Simplifying Assumption: Use only the previous word

P(office | about fifteen minutes from) $\approx$ P(office | from)

Markov Assumption

Simplifying Assumption: Use only the previous word

P(office | about fifteen minutes from) $\approx$ P(office | from)

Or the couple previous words

P(office | about fifteen minutes from) $\approx$ P(office | minutes from)

Markov Assumption

More Formally: kth order Markov Model

Chain Rule:

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

Markov Assumption

More Formally: kth order Markov Model

Chain Rule:

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

Using Markov Assumption: only k previous words

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

Markov Assumption

More Formally: kth order Markov Model

Chain Rule:

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

Using Markov Assumption: only k previous words

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

We approximate each component in the product

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

N-Gram Models

P(office | about fifteen minutes from)

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

N-Gram Models

P(office | about fifteen minutes from)

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

- Unigram: P(office)
- Bigram: P(office | from)
- Trigram: P(office | minutes from)

N-Gram Models

P(office | about fifteen minutes from)

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

- Unigram: P(office)
- Bigram: P(office | from)
- Trigram: P(office | minutes from)

Markov model and Language Model

N-Gram Models

P(office | about fifteen minutes from)

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

- Unigram: P(office)
- Bigram: P(office | from)
- Trigram: P(office | minutes from)

Markov model and Language Model

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

N-Gram Models

- We can extend to trigrams, 4-grams, 5-grams
- In general, an insufficient model of language:

N-Gram Models

- 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**."

N-Gram Models

- 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

Estimating N-grams probabilities

Estimating N-grams probabilities

Maximum Likelihood Estimate

Value that makes the observed data the "most probable"

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

Estimating N-grams probabilities

Maximum Likelihood Estimate

Value that makes the observed data the "most probable"

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

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

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>

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>) =
P(</s>|here) =
P(would | I) =
P(here | am) =
P(know | like) =

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

Bigram counts from 9222 Restaurant Sentences

	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

Computing bigram probabilities

Normlize by unigrams

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

Computing bigram probabilities

Normlize by unigrams

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

Bigram Probabilities

	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

Computing Sentence Probabilities

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

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

Computing Sentence Probabilities

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

= P(I | <s>) x P(want | I) x P(english | want) x P(food | english) x P(</s> | food)
= 0.000031

What knowledge does n-gram represent?

- P(english|want) = .0011
- P(chinese|want) = .0065
- P(to|want) = .66
- P(eat | to) = .28
- P(food | to) = 0
- P(want | spend) = 0
- P (i | <s>) = .25

Practical Issues

Everything in log space

- Avoids underflow

Practical Issues

Everything in log space

- Avoids underflow
- Adding is faster than multiplying

Practical Issues

Everything in log space

- Avoids underflow
- Adding is faster than multiplying

$$log(p_1 \times p_2 \times p_3 \times p_4) = logp_1 + logp_2 + logp_3 + logp_4$$

Handling zeros

Use smoothing

Language Modeling Toolkit

SRILM

http://www.speech.sri.com/projects/srilm/

Google N-grams

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
`http://googleresearch.blogspot.in/2006/08/all-our-n-gram-are-belong-to-you.html`

Example from the 4-gram data

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

Google books Ngram Data