# CS565: Intelligent Systems and Interfaces

Lecture: Language Modeling

28th Jan, 2016

Semester: Jan – May 2016

Ashish Anand

IIT Guwahati

# Recap and Moving Forward

- Until Now
  - Sentence segmentation, Tokenization
  - Collocation
- Next
  - Language Modeling: Generative model of language

# Understanding Language Modeling

# Language Modeling (LM)

- Assigning probabilities to
  - possible next words
  - sequence of words
- Word Prediction
  - I am attending …..
  - What is the difference between generative and ….
  - Need to estimate $P(w_n | w_1, w_2, w_3, \ldots, w_{n-1})$

# Applications

- Speech Recognition
  - *P(I saw a van) > P(I saw 7)*

- Spell Correction
  - *P( study was conducted **by** students) > P(study was conducted **be** students)*

- Other Application
  - Machine Translation, Summarization, Augmentative communication system etc.

- Need to compute $P(w_1, w_2, w_3, \ldots, w_n)$

# Defining LM Formally

- We consider a *vocabulary,* a finite set denoted as $\mathcal{V}$, and a function

$P(w_1, w_2, w_3, .... , w_n)$, such that
  - For any $<w_1, w_2, w_3, .... , w_n> \in \mathcal{V}^\dagger$, $p(w1, w2, w3, ..., wn) \geq 0$
  - $\Sigma\ p(w1, w2, w3, ..., wn) = 1$,

*where* $\mathcal{V}^\dagger : \{S = w_1 w_2 w_3 ....w_n \mid w_i \in \mathcal{V}\}$.

# How to compute $P(w_1, w_2, .., w_n)$

- Our task is to compute

  P(I, am, fascinated, with, recent, advances, in, AI)

- Chain Rule
  - $P(w_1, w_2, w_3, \ldots., w_n) = P(w_1)\, P(w_2|w_1)\, P(w_3|w_1, w_2) \ldots.\, P(w_n|w_1,.., w_{n-1})$

# Estimating $P(w_n | w_1,.., w_{n-1})$

- Could we just count and divide?

$$P(eat|I\ want\ to) = \frac{count(I\ want\ to\ eat)}{count(I\ want\ to)}$$

- What is the problem here?

# Estimating $P(w_n | w_1,.., w_{n-1})$

- Too many possible sentences
- Data sparseness
- Poor *generalizability*

# Markov Assumption

- Simplifying assumption:

$$P(eat \mid I\ want\ to) \sim P(eat \mid to)$$

or

$$P(eat \mid I\ want\ to) \sim P(eat \mid want\ to)$$

# Markov Assumption

$$P(w_1, w_2, w_3, \ldots., w_n) \sim \prod_i P(w_i | w_{i-k}, \ldots., w_{i-1})$$

i.e., Each component in the product is getting approximated by Markov assumption

$$P(w_i | w_1, w_2, w_3, \ldots., w_{i-1}) \sim P(w_i \mid w_{i-k}, \ldots., w_{i-1})$$

# N-gram Models

- Unigram: Simplest Model (does not depend on anything)

$$P(w_1, w_2, w_3, \ldots., w_n) \sim \prod_i P(wi)$$

- Bigram Model (1st Order Markov model)

$$P(w_1, w_2, w_3, \ldots., w_n) \sim \prod_i P(w_i|w_{i-1})$$

- Trigram Model (2nd order Markov model)

$$P(w_1, w_2, w_3, \ldots., w_n) \sim \prod_i P(w_i|w_{i-1}, w_{i-2})$$

# N-gram Model: Issue

- Long-distance dependencies

*“The computer which I had just put into the lab on the fifth floor crashed”*

# Estimating the Probabilities

# Data

- Training
- Development
- Test

# Maximum Likelihood Estimate

- Unigram

$$P(w_i) = \frac{c(w_i)}{K} \;;$$

$$K\text{: } \textit{Total number of } \boldsymbol{tokens} \textit{ in training set}$$

- Bigram

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

- N-Gram

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{c(w_{n-N+1}^{n-1}\; w_n)}{c(w_{n-N+1}^{n-1})}$$

# Bigram Probabilities

| | | | |
|---|---|---|---|
| eat on | .16 | eat Thai | .03 |
| eat some | .06 | eat breakfast | .03 |
| eat lunch | .06 | eat in | .02 |
| eat dinner | .05 | eat Chinese | .02 |
| eat at | .04 | eat Mexican | .02 |
| eat a | .04 | eat tomorrow | .01 |
| eat Indian | .04 | eat dessert | .007 |
| eat today | .03 | eat British | .001 |

A fragment of bigram probabilities from the *Berkeley Restaurant Project* showing most likely word to follow *eat*

Source: Figure 6.2: Page 225, SLP

# Computing probability of a sentence

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| <s> I | .25 | I want | .32 | want to | .65 | to eat | .26 | British food | .60 |
| <s> I'd | .06 | I would | .29 | want a | .05 | to have | .14 | British restaurant | .15 |
| <s> Tell | .04 | I don't | .08 | want some | .04 | to spend | .09 | British cuisine | .01 |
| <s> I'm | .02 | I have | .04 | want thai | .01 | to be | .02 | British lunch | .01 |

**Figure 6.3** More fragments from the bigram grammar from the Berkeley Restaurant Project.

*P (<s> I want to eat British food </s>) = P(I|<s>) P(want|I) P(to|want) P(eat|to) P(British|eat) P(food|British) P(</s>|food)*

# Practical Issue

- Avoiding underflow
  - Work in log space

$$\log(p_1, p_2, p_3, p_4) = \sum \log p_i$$

# Do we see any problem here?

- Bigram counts

| | I | want | to | eat | Chinese | food | lunch |
|---|---|---|---|---|---|---|---|
| I | 8 | 1087 | 0 | 13 | 0 | 0 | 0 |
| want | 3 | 0 | 786 | 0 | 6 | 8 | 6 |
| to | 3 | 0 | 10 | 860 | 3 | 0 | 12 |
| eat | 0 | 0 | 2 | 0 | 19 | 2 | 52 |
| Chinese | 2 | 0 | 0 | 0 | 0 | 120 | 1 |
| food | 19 | 0 | 17 | 0 | 0 | 0 | 0 |
| lunch | 4 | 0 | 0 | 0 | 0 | 1 | 0 |

# Problem with MLE

| | I | want | to | eat | Chinese | food | lunch |
|---|---|---|---|---|---|---|---|
| I | .0023 | .32 | 0 | .0038 | 0 | 0 | 0 |
| want | .0025 | 0 | .65 | 0 | .0049 | .0066 | .0049 |
| to | .00092 | 0 | .0031 | .26 | .00092 | 0 | .0037 |
| eat | 0 | 0 | .0021 | 0 | .020 | .0021 | .055 |
| Chinese | .0094 | 0 | 0 | 0 | 0 | .56 | .0047 |
| food | .013 | 0 | .011 | 0 | 0 | 0 | 0 |
| lunch | .0087 | 0 | 0 | 0 | 0 | .0022 | 0 |

# Problem with MLE

- Shakespeare as corpus
  - N = 884,647 tokens; V = 29066
  - Bigrams in corpus: 300,000 [Possible: V^2]
  - 99.96% of possible bigrams never seen.
  - What will happen with higher order?

# Problem: Continued

- Works well if test corpus is very similar to training, which is not generally the case.
- Training Set

  …… denied the allegations

  …… denied the reports

  …… denied the claims

  …… denied the request
- Test Set

  …. denied the offer

  …. denied the loan

P("offer" | denied the) = 0

# Generalization and taking care of zero or low probability

- Smoothing Techniques: Next Lecture

- Evaluation of N-Gram Models: Next Lecture

# Reference

- Collins Lecture on Language Modeling
- Chapter 6-6.2: *Speech and Language Processing [SLP]*, Jurafsky and Martin