SI425 : NLP

Set 3

Language Models

Fall 2017 : Chambers

Language Modeling

- Which sentence is most likely (most probable)?

I saw this dog running across the street.

Saw dog this I running across street the.

Why? *You have a language model in your head.*

P(“I saw this”) >> P(“saw dog this”)

Language Modeling

- Compute $P(w_1, w_2, w_3, w_4, w_5, \ldots, w_n)$
  - the probability of a sequence
- Compute $P(w_5|w_1, w_2, w_3, w_4)$
  - the probability of a word given some previous words

- The model that computes P(W) is the **language model**.
- A better term for this would be “The Grammar”
  - “Language model” or LM is standard

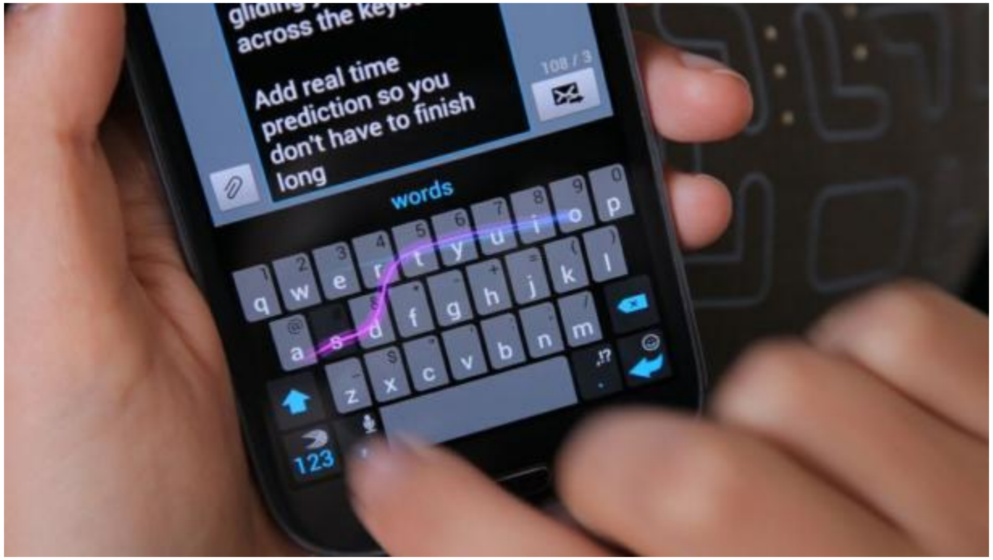
LMs: “fill in the blank”

- Can also think of this as a “fill in the blank” problem.

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

“He picked up the bat and hit the _____”

Ball? Poetry?



How do we count words?

“They picnicked by the pool then lay back on the grass and looked at the stars”

- 16 tokens
- 14 types

- The Brown Corpus (1992): a big corpus of English text
  - 583 million wordform tokens
  - 293,181 wordform types

- **N** = number of tokens
- **V** = vocabulary = number of types
- General wisdom: **V** > O(sqrt(**N**))

Computing P(W)

- How to compute this?

  P("The other day I was walking along and saw a lizard")

- Compute the joint probability of its tokens *in order*:

  P("The","other","day","I","was","walking","along","and","saw","a","lizard")

- Rely on the Chain Rule of Probability

The Chain Rule of Probability

- Recall the definition of conditional probabilities

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

- Rewriting:

$$P(A, B) = P(A \mid B)P(B)$$

- More generally:

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

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

The Chain Rule for a sentence

$$\begin{aligned} P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2)\ldots P(w_n|w_1^{n-1}) \\ &= \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \end{aligned}$$

- P("the big red dog was") = ???

P(the) * P(big|the) * P(red|the big) *
P(dog|the big red) * P(was|the big red dog) = ???

Very easy to estimate

How to estimate?

- P(the | its water is so transparent that)

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

Unfortunately

- There are a lot of possible sentences.

- We'll never be able to get enough data to compute the statistics for these long prefixes.

P(lizard | the,other,day,I,was,walking,along,and,saw,a)

Markov Assumption

- Make a simplifying assumption

  P(lizard | the,other,day,I,was,walking,along,and,saw,a) **=**

  P(lizard | a)

- Or maybe

  P(lizard | the,other,day,I,was,walking,along,and,saw,a) **=**

  P(lizard | saw, a)

Markov Assumption

- So for each component in the product, replace with the approximation (assuming a prefix of N)

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-N+1}^{n-1})$$

- Bigram version

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-1})$$

N-gram Terminology

- **Unigrams**: single words
- **Bigrams**: pairs of words
- **Trigrams**: three word phrases
- 4-grams, 5-grams, 6-grams, etc.

Attention!
We don't include **<s>** as a token. It is just context.
But we do count **</s>** as a token.

"I saw a lizard yesterday"

Unigrams	Bigrams	Trigrams
I	<s> I	<s> <s> I
saw	I saw	<s> I saw
a	saw a	I saw a
lizard	a lizard	saw a lizard
yesterday	lizard yesterday	a lizard yesterday
</s>	yesterday </s>	lizard yesterday </s>

Estimating bigram probabilities

- The Maximum Likelihood Estimate

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

Bigram language model: what counts do I have to keep track of??

An example

<s> I am Sam </s>

<s> Sam I am </s>

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

$P(\text{I} \mid \text{<s>}) = \frac{2}{3} = .66 \qquad P(\text{Sam} \mid \text{<s>}) = \frac{1}{3} = .33 \qquad P(\text{am} \mid \text{I}) = \frac{2}{3} = .33$

$P(\text{</s>} \mid \text{Sam}) = \frac{1}{2} = 0.5 \qquad P(\text{<s>} \mid \text{Sam}) = \frac{1}{2} = 0.5 \qquad P(\text{Sam} \mid \text{am}) = \frac{1}{2} = .5$

$P(\text{do} \mid \text{I}) = \frac{1}{3} = .33$

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

- This is the Maximum Likelihood Estimate, because it is the one which maximizes P(text-data | model)

Maximum Likelihood Estimates

- The MLE of a parameter in a model M from a training set T
  - …is the estimate that maximizes the likelihood of the training set T given the model M

- “Chinese” occurs 400 times in a corpus
- What is the probability that a random word from another text will be “Chinese”?
- MLE estimate is 400/1,000,000 = .004
  - This may be a bad estimate for some other corpus
  - *But it is the **estimate** that makes it **most likely** that “Chinese” will occur 400 times in a million word corpus.*

Example: Berkeley Restaurant Project

- can you tell me about any good cantonese restaurants close by
- mid priced thai food is what i’m looking for
- tell me about chez panisse
- can you give me a listing of the kinds of food that are available
- i’m looking for a good place to eat breakfast
- when is caffe venezia open during the day

Raw bigram counts

- Out of 9222 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

Raw bigram probabilities

- Normalize by unigram counts:

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

- Result:

	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

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)

= .24 x .33 x .0011 x 0.5 x 0.68

=.000031

Unknown words

P(They eat **lutefisk** in Norway) = 0.0

If **lutefisk** was never seen, then the entire sentence is 0!

- Closed Vocabulary Task
  - We know all the words in advanced
  - Vocabulary V is fixed
- Open Vocabulary Task
  - You typically don't know the vocabulary
  - Out Of Vocabulary = OOV words

Unknown words: Fixed lexicon solution

- Create a fixed lexicon L of size V
- Create an unknown word token **<UNK>**

- Training
  - At text normalization phase, any training word not in L changed to <UNK>
  - Train its probabilities like a normal word
- At decoding time
  - Use <UNK> probabilities for any word not in training

Unknown words: A Simplistic Approach

- Count all tokens in your training set.
- Create an “unknown” token **<UNK>**

- Assign probability P(<UNK>) = 1 / (N+1)
- All other tokens receive P(word) = C(word) / (N+1)

- During testing, any new word not in the vocabulary receives P(<UNK>).

Evaluate

- I counted a bunch of words. But is my language model any good?

1. Auto-generate sentences
2. Perplexity
3. Word-Error Rate

The Shannon Visualization Method

- Generate random sentences:
- Choose a random bigram “<s> w” according to its probability
- Now choose a random bigram “w x” according to its probability
- And so on until we randomly choose “</s>”
- Then string the words together
- <s> I

    I want

    want to

    to eat

    eat Chinese

    Chinese food

    food </s>

Unigram	• To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have • Every enter now severally so, let • Hill he late speaks; or! a more to leg less first you enter • Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like
Bigram	• What means, sir. I confess she? then all sorts, he is trim, captain. •Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow. •What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman? •Enter Menenius, if it so many good direction found'st thou art a strong upon command of fear not a liberal largess given away, Falstaff! Exeunt
Trigram	• Sweet prince, Falstaff shall die. Harry of Monmouth's grave. • This shall forbid it should be branded, if renown made it empty. • Indeed the duke; and had a very good friend. • Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.
Quadrigram	• King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in; • Will you not tell me who I am? • It cannot be but so. • Indeed the short and the long. Marry, 'tis a noble Lepidus.

Evaluation

- We learned probabilities from a **training set**.
- Look at the model's performance on some new data
  - This is a **test set**. A dataset different than our training set

- Then we need an **evaluation metric** to tell us how well our model is doing on the test set.
- One such metric is **perplexity**

Perplexity

- Perplexity is the probability of the test set (assigned by the language model), normalized by the number of words:

$$\begin{aligned} PP(W) &= P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}} \\ &= \sqrt[N]{\frac{1}{P(w_1 w_2 \ldots w_N}}) \end{aligned}$$

- Chain rule:

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_1 \ldots w_{i-1})}}$$

- For bigrams:

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_{i-1})}}$$

- Minimizing perplexity is the same as maximizing probability
  - **The best language model is one that best predicts an unseen test set**

Lower perplexity = better model

- Training 38 million words, test 1.5 million words, WSJ

N-gram Order	Unigram	Bigram	Trigram
Perplexity	962	170	109

- Begin the lab! Make bigram and trigram models!