Text: $P(w_1, w_2, w_3 \ldots w_N) \Rightarrow .0000005$

Probabilistic Language Models

- Today's goal: assign a probability to a sentence or text

Why?

- Machine Translation:
  - P(**high** winds tonite) > P(**large** winds tonite)
- Spell Correction
  - The office is about fifteen **minuets** from my house
    - P(about fifteen **minutes** from) > P(about fifteen **minuets** from)
- Speech Recognition
  - P(I saw a van) >> P(eyes awe of an)
- + Summarization, question-answering, etc., etc.!!

9/12/17 UMass CS 585, Intro to NLP
Slides: SLP website

Probabilistic Language Modeling

- Goal: compute the probability of a sentence or sequence of words: $w_i \in V$ (vocabulary)

  $P(W) = P(w_1,w_2,w_3,w_4,w_5 \ldots w_n)$

- Related task: probability of an upcoming word:

  $P(w_5|w_1,w_2,w_3,w_4)$

- A model that computes either of these:

  $P(W)$ or $P(w_n|w_1,w_2 \ldots w_{n-1})$ is called a **language model**.

- Better: **the grammar** But **language model** or **LM** is standard

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

$P(\underline{A}\,\underline{BC}) = P(A|BC)\,P(BC)$

$= P(A|BC)\,P(B|C)\,P(C)$

How to compute P(W)

- How to compute this joint probability:

  $P(w_4 = \text{so} \,|\, (w_1, w_2, w_3) = (\text{its, water, is}))$

  - P(its, water, is, so, transparent, that)

- Intuition: let's rely on the Chain Rule of Probability

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

Cond. Prob. Dist over V

(<START>, its, water, is . . .)

$$P(w_1 \ldots w_N) = \prod_i P(w_i \mid w_1 \ldots w_{i-1})$$
$$\log P(\ldots) = \sum_i \log P(w_i \mid w_1 \ldots w_{i-1})$$

Intuition of Perplexity

- The Shannon Game:
  - How well can we predict the next word?

    I always order pizza with cheese and ____

    The 33rd President of the US was ____

    I saw a ____

  - Unigrams are terrible at this game. (Why?)
- A better model of a text
  - is one which assigns a higher probability to the word that actually occurs

mushrooms 0.1
pepperoni 0.1
anchovies 0.01
....
fried rice 0.0001
....
and 1e-100

N-Gram LMs

How to estimate these probabilities

- Could we just count and divide?

Bad: Large/long contexts

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

$$\frac{Count(\text{its water is so transparent that the})}{Count(\text{its water is so transparent that})} = \frac{3}{100{,}000}$$

- No! Too many possible sentences!
- We'll never see enough data for estimating these

Phrase len $K \Rightarrow |V|^K$ possible strings

Markov Assumption

Andrei Markov

- Simplifying assumption:

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

"Length 1 history MM"

- Or maybe

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

"Length 2 history MM"

"Second-order MM"

Markov Assumption

size k

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

- In other words, we approximate each component in the product

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

Simplest case: Unigram model

$$P(w_1 w_2 \ldots w_n) \approx \prod_i P(w_i)$$

Some automatically generated sentences from a unigram model

```
fifth, an, of, futures, the, an, incorporated, a,
a, the, inflation, most, dollars, quarter, in, is,
mass

thrift, did, eighty, said, hard, 'm, july, bullish

that, or, limited, the
```

Bigram model

- Condition on the previous word:

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

$P(? \mid \text{one})$

```
texaco, rose, one, in, this, issue, is, pursuing, growth, in,
a, boiler, house, said, mr., gurria, mexico, 's, motion,
control, proposal, without, permission, from, five, hundred,
fifty, five, yen

outside, new, car, parking, lot, of, the, agreement, reached

this, would, be, a, record, november
```

N-gram models

- We can extend to trigrams, 4-grams, 5-grams
- In general this is an insufficient model of language
  - because language has **long-distance dependencies**:

"The computer which I had just put into the machine room on the fifth floor crashed."

- But we can often get away with N-gram models

Estimating bigram probabilities

- The Maximum Likelihood Estimate

from training corpus

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

$$p(\text{car} \mid \text{the}) = \frac{c[\text{"the car"}]}{c[\text{"the"}]}$$

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

Sparsity! Is bad

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