Probabilistic Language Models

- Today’s goal: assign a probability to a sentence
- Machine Translation:
  - \( P(\text{high winds tonite}) > P(\text{large winds tonite}) \)
- Spell Correction
  - The office is about fifteen \textit{minuets} from my house
  - \( P(\text{about fifteen minutes from}) > P(\text{about fifteen minuets from}) \)
- Speech Recognition
  - \( P(\underline{I \text{ saw a van}}) >> P(\underline{eyes awe of an}) \)
- + 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:
  \[ P(W) = P(w_1, w_2, w_3, w_4, w_5 \ldots w_n) \]

- 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:
  \[ P(W) \quad \text{or} \quad P(w_n | w_1, w_2 \ldots w_{n-1}) \]
  is called a language model.

- Better: the grammar   But language model or LM is standard
How to compute $P(W)$

- How to compute this joint probability:
  
  $P(\text{its, water, is, so, transparent, that})$

- Intuition: let's rely on the Chain Rule of Probability
  
  $P(w_1, \ldots, w_N) = \prod_{i=1}^{N} P(w_i | w_1, w_2, \ldots, w_{i-1})$

  $\uparrow$

  Cond. Prob. Dist over $\{\text{START}, \text{its, water, is, \ldots}\}$
\[ P(w_1 \ldots w_n) = \prod_i P(w_i | w_{i-1}) \]
\[ \log P(w_1 \ldots w_n) = \sum_i \log P(w_i | w_{i-1}) \]

**Intuition of Perplexity**

- The Shannon Game:
  - How well can we predict the next word?
    - I always order pizza with cheese and ____
    - The 33\textsuperscript{rd} 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
How to estimate these probabilities

- Could we just count and divide?

\[
P(\text{its water is so transparent that}) = \frac{\text{Count(its water is so transparent that the)}}{\text{Count(its water is so transparent that)}} = \frac{5}{100,000}
\]

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

\[
\text{Phrase len } K \Rightarrow |V|^K \text{ possible strings}
\]
Markov Assumption

• Simplifying assumption:

$$P(\text{the lits water is so transparent that}) \approx P(\text{the | that})$$

• Or maybe

$$P(\text{the lits water is so transparent that}) \approx P(\text{the | transparent that})$$
Markov Assumption

\[ P(w_1 w_2 \ldots w_n) \approx \prod_{i} P(w_i | w_{i-k} \ldots w_{i-1}) \]

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

\[ P(w_i | w_1 w_2 \ldots w_{i-1}) \approx P(w_i | w_{i-k} \ldots w_{i-1}) \]
Simplest case: Unigram model

\[ P(w_1w_2\ldots w_n) \approx \prod_i P(w_i) \]

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:

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

- 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

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

\[ P(\text{can} \mid \text{the}) = \frac{\frac{\text{count}(\text{can} \mid \text{the})}{\text{count}(\text{the})}}{\text{count}(\text{the})} \]

\[ P(w_i \mid w_{i-1}) = \frac{\text{count}(w_{i-1}, w_i)}{\text{c}(w_{i-1})} \]
### Raw bigram counts

- Out of 9222 sentences

<table>
<thead>
<tr>
<th></th>
<th>i</th>
<th>want</th>
<th>to</th>
<th>eat</th>
<th>chinese</th>
<th>food</th>
<th>lunch</th>
<th>spend</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>5</td>
<td>827</td>
<td>0</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>want</td>
<td>2</td>
<td>0</td>
<td>608</td>
<td>1</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>to</td>
<td>2</td>
<td>0</td>
<td>4</td>
<td>686</td>
<td>2</td>
<td>0</td>
<td>6</td>
<td>211</td>
</tr>
<tr>
<td>eat</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>16</td>
<td>2</td>
<td>42</td>
<td>0</td>
</tr>
<tr>
<td>chinese</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>82</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>food</td>
<td>15</td>
<td>0</td>
<td>15</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>lunch</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>spend</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>