Automatic Speech Recognition (CS753)

Lecture 11: Language Models (Part I)
Assignment 2 Schedule
(Moodle responses)

Votes

<table>
<thead>
<tr>
<th>Dates</th>
<th>Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aug 31/Sept 18</td>
<td>9</td>
</tr>
<tr>
<td>Sept 7/Sept 25</td>
<td>20</td>
</tr>
<tr>
<td>Sept 18/Oct 5</td>
<td>8</td>
</tr>
</tbody>
</table>
So far, acoustic models...

Acoustic Models

Context Transducer

Pronunciation Model

Language Model

Word Sequence

Acoustic Indices

Triphones

Monophones

Words

b+ae+n

b+iy+n

k+ae+n
Next, language models

- Language models
  - provide information about word reordering
    \[ \Pr(\text{“she class taught a”}) < \Pr(\text{“she taught a class”}) \]
  - provide information about the most likely next word
    \[ \Pr(\text{“she taught a class”}) > \Pr(\text{“she taught a speech”}) \]
Application of language models

- Speech recognition
  - $\Pr(\text{“she taught a class”}) > \Pr(\text{“sheet or tuck lass”})$
- Machine translation
- Handwriting recognition/Optical character recognition
- Spelling correction of sentences
- Summarization, dialog generation, information retrieval, etc.
Popular Language Modelling Toolkits

- SRILM Toolkit:
  
  http://www.speech.sri.com/projects/srilm/

- KenLM Toolkit:
  
  https://kheafield.com/code/kenlm/

- OpenGrm NGram Library:
  
  http://opengrm.org/
Introduction to probabilistic LMs
Probabilistic or Statistical Language Models

- Given a word sequence, \( W = \{w_1, \ldots, w_n\} \), what is \( \Pr(W) \)?

- Decompose \( \Pr(W) \) using the chain rule:

\[
\Pr(w_1, w_2, \ldots, w_{n-1}, w_n) = \Pr(w_1) \Pr(w_2|w_1) \Pr(w_3|w_1, w_2) \ldots \Pr(w_n|w_1, \ldots, w_{n-1})
\]

- Sparse data with long word contexts: How do we estimate the probabilities \( \Pr(w_n|w_1, \ldots, w_{n-1}) \)?
Estimating word probabilities

• Accumulate counts of words and word contexts

• Compute normalised counts to get next-word probabilities

• E.g. $\Pr(\text{“class | she taught a”})$

$$= \frac{\pi(\text{“she taught a class”})}{\pi(\text{“she taught a”})}$$

where $\pi(\ldots)$ refers to counts derived from a large English text corpus

• What is the obvious limitation here? We’ll never see enough data
Simplifying Markov Assumption

- Markov chain:
  - Limited memory of previous word history: Only last $m$ words are included

- 1-order language model (or bigram model)
  $$\Pr(w_1, w_2, \ldots, w_{n-1}, w_n) \approx \Pr(w_1) \Pr(w_2|w_1) \Pr(w_3|w_2) \ldots \Pr(w_n|w_{n-1})$$

- 2-order language model (or trigram model)
  $$\Pr(w_1, w_2, \ldots, w_{n-1}, w_n) \approx \Pr(w_1) \Pr(w_2|w_1) \Pr(w_3|w_1, w_2) \ldots \Pr(w_n|w_{n-2}, w_{n-1})$$

- Ngram model is an $N$-1th order Markov model
Estimating Ngram Probabilities

• Maximum Likelihood Estimates

  • Unigram model

    \[ \Pr_{ML}(w_1) = \frac{\pi(w_1)}{\sum_i \pi(w_i)} \]

  • Bigram model

    \[ \Pr_{ML}(w_2|w_1) = \frac{\pi(w_1, w_2)}{\sum_i \pi(w_1, w_i)} \]
Example

The dog chased a cat
The cat chased away a mouse
The mouse eats cheese

What is \( \text{Pr}("\text{The cat chased a mouse}") \) using a bigram model?

\[
\text{Pr}("<s> \text{The cat chased a mouse } </s>") = \\
\text{Pr}("\text{The } <s>\) \cdot \text{Pr}("\text{cat } \text{The}\) \cdot \text{Pr}("\text{chased } \text{cat}\) \cdot \text{Pr}("\text{a } \text{chased}\) \cdot \text{Pr}("\text{mouse } \text{a}\) \cdot \text{Pr}("</s> \text{mouse}\) = \\
3/3 \cdot 1/3 \cdot 1/2 \cdot 1/2 \cdot 1/2 \cdot 1/2 = 1/48
\]
Example

The dog chased a cat
The cat chased away a mouse
The mouse eats cheese

What is \( \text{Pr}(\text{"The dog eats cheese"}) \) using a bigram model?

\[
\text{Pr}(\text{"<s> The dog eats cheese </s>"}) = \\
\text{Pr}(\text{"The|<s>"}) \cdot \text{Pr}(\text{"dog|The"}) \cdot \text{Pr}(\text{"eats|dog"}) \cdot \text{Pr}(\text{"cheese|eats"}) \cdot \text{Pr}(\text{"</s>|cheese"}) = \\
3/3 \cdot 1/3 \cdot 0/1 \cdot 1/1 \cdot 1/1 = 0! \quad \text{Due to unseen bigrams}
\]

How do we deal with unseen bigrams? We’ll come back to it.
Open vs. closed vocabulary task

- Closed vocabulary task: Use a fixed vocabulary, V. We know all the words in advance.

- More realistic setting, we don’t know all the words in advance. Open vocabulary task. Encounter out-of-vocabulary (OOV) words during test time.

- Create an unknown word: <UNK>
  - Estimating <UNK> probabilities: Determine a vocabulary V. Change all words in the training set not in V to <UNK>
  - Now train its probabilities like a regular word
  - At test time, use <UNK> probabilities for words not in training
Evaluating Language Models

- Extrinsic evaluation:
  - To compare Ngram models A and B, use both within a specific speech recognition system (keeping all other components the same)
  - Compare word error rates (WERs) for A and B
  - Time-consuming process!
Intrinsic Evaluation

• Evaluate the language model in a standalone manner

• How likely does the model consider the text in a test set?

• How closely does the model approximate the actual (test set) distribution?

• Same measure can be used to address both questions — perplexity!
Measures of LM quality

• How likely does the model consider the text in a test set?

• How closely does the model approximate the actual (test set) distribution?

• Same measure can be used to address both questions — perplexity!
Perplexity (I)

• How likely does the model consider the text in a test set?

  • Perplexity(test) = \(1/Pr_{\text{model}}[\text{text}]\)

• Normalized by text length:

  • Perplexity(test) = \((1/Pr_{\text{model}}[\text{text}])^{1/N}\) where \(N = \text{number of tokens in test}\)

• e.g. If model predicts i.i.d. words from a dictionary of size \(L\), per word perplexity = \((1/(1/L)^{N})^{1/N} = L\)
Intuition for Perplexity

- Shannon’s guessing game builds intuition for perplexity

- What is the surprisal factor in predicting the next word?

- At the stall, I had tea and _________ biscuits 0.1
  samosa 0.1
  coffee 0.01
  rice 0.001
  but 0.000000000001

- A better language model would assign a higher probability to the actual word that fills the blank (and hence lead to lesser surprisal/perplexity)
Measures of LM quality

• How likely does the model consider the text in a test set?

• How closely does the model approximate the actual (test set) distribution?

• Same measure can be used to address both questions — perplexity!
Perplexity (II)

- How closely does the model approximate the actual (test set) distribution?

  - KL-divergence between two distributions $X$ and $Y$
    \[
    D_{KL}(X||Y) = \sum_{\sigma} Pr_X[\sigma] \log \left( \frac{Pr_X[\sigma]}{Pr_Y[\sigma]} \right)
    \]
  
  - Equals zero iff $X = Y$; Otherwise, positive

- How to measure $D_{KL}(X||Y)$? We don’t know $X$!

  - $D_{KL}(X||Y) = \sum_{\sigma} Pr_X[\sigma] \log \left( \frac{1}{Pr_Y[\sigma]} \right) - H(X)$
    where $H(X) = -\sum_{\sigma} Pr_X[\sigma] \log Pr_X[\sigma]$

  - Empirical cross entropy:
    \[
    \frac{1}{|test|} \sum_{\sigma \in test} \log \left( \frac{1}{Pr_y[\sigma]} \right)
    \]
Perplexity vs. Empirical Cross Entropy

• Empirical Cross Entropy (ECE)

\[
\frac{1}{|\text{#sents}|} \sum_{\sigma \in \text{test}} \log\left( \frac{1}{\Pr_{\text{model}}[\sigma]} \right)
\]

• Normalized Empirical Cross Entropy = ECE/(avg. length) =

\[
\frac{1}{|\text{#words}/\text{#sents}|} \frac{1}{|\text{#sents}|} \sum_{\sigma \in \text{test}} \log\left( \frac{1}{\Pr_{\text{model}}[\sigma]} \right) = \\
\frac{1}{N} \sum_{\sigma} \log\left( \frac{1}{\Pr_{\text{model}}[\sigma]} \right)
\]

where \( N = \text{#words} \)

• How does \( \frac{1}{N} \sum_{\sigma} \log\left( \frac{1}{\Pr_{\text{model}}[\sigma]} \right) \) relate to perplexity?
Perplexity vs. Empirical Cross Entropy

\[
\log(\text{perplexity}) = \frac{1}{N} \log \frac{1}{\Pr[\text{test}]}
\]

\[
= \frac{1}{N} \log \prod_{\sigma} \left( \frac{1}{\Pr_{\text{model}}[\sigma]} \right)
\]

\[
= \frac{1}{N} \sum_{\sigma} \log \left( \frac{1}{\Pr_{\text{model}}[\sigma]} \right)
\]

Thus, perplexity = \exp^{\text{(normalized cross entropy)}}

Example perplexities for Ngram models trained on WSJ (80M words):

Unigram: 962, Bigram: 170, Trigram: 109
Introduction to smoothing of LMs
Recall example

The dog chased a cat
The cat chased away a mouse
The mouse eats cheese

What is Pr(“The dog eats cheese”)?

Pr(“<s> The dog eats cheese </s>”) =

Pr(“The|<s>”) \cdot Pr(“dog|The”) \cdot Pr(“eats|dog”) \cdot Pr(“cheese|eats”) \cdot Pr(“</s>|cheese”)

= 3/3 \cdot 1/3 \cdot 0/1 \cdot 1/1 \cdot 1/1 = 0! \quad \text{Due to unseen bigrams}
Unseen Ngrams

- Even with MLE estimates based on counts from large text corpora, there will be many unseen bigrams/trigrams that never appear in the corpus.

- If any unseen Ngram appears in a test sentence, the sentence will be assigned probability 0.

- Problem with MLE estimates: maximises the likelihood of the observed data by assuming anything unseen cannot happen and overfits to the training data.

- **Smoothing methods:** Reserve some probability mass to Ngrams that don’t occur in the training corpus.
Add-one (Laplace) smoothing

Simple idea: Add one to all bigram counts. That means,

\[
\Pr_{ML}(w_i|w_{i-1}) = \frac{\pi(w_{i-1}, w_i)}{\pi(w_{i-1})}
\]

becomes

\[
\Pr_{Lap}(w_i|w_{i-1}) = \frac{\pi(w_{i-1}, w_i) + 1}{\pi(w_{i-1})}
\]

Correct?
Add-one (Laplace) smoothing

Simple idea: Add one to all bigram counts. That means,

$$\Pr_{ML}(w_i|w_{i-1}) = \frac{\pi(w_{i-1}, w_i)}{\pi(w_{i-1})}$$

becomes

$$\Pr_{Lap}(w_i|w_{i-1}) = \frac{\pi(w_{i-1}, w_i) + 1}{\pi(w_{i-1})}$$

No, $\Sigma w_i \Pr_{Lap}(w_i|w_{i-1})$ must equal 1. Change denominator s.t.

$$\sum_{w_i} \frac{\pi(w_{i-1}, w_i) + 1}{\pi(w_{i-1}) + x} = 1$$

Solve for $x$: $x = V$ where $V$ is the vocabulary size
Add-one (Laplace) smoothing

Simple idea: Add one to all bigram counts. That means,

$$\Pr_{ML}(w_i|w_{i-1}) = \frac{\pi(w_{i-1}, w_i)}{\pi(w_{i-1})}$$

becomes

$$\Pr_{Lap}(w_i|w_{i-1}) = \frac{\pi(w_{i-1}, w_i) + 1}{\pi(w_{i-1}) + V}$$

where $V$ is the vocabulary size.
## Example: Bigram counts

<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><strong>i</strong></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>

### No smoothing

### Laplace (Add-one) smoothing
Example: Bigram probabilities

<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>0.002</td>
<td>0.33</td>
<td>0</td>
<td>0.0036</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.00079</td>
</tr>
<tr>
<td>want</td>
<td>0.0022</td>
<td>0</td>
<td>0.66</td>
<td>0.0011</td>
<td>0.0065</td>
<td>0.0065</td>
<td>0.0054</td>
<td>0.0011</td>
</tr>
<tr>
<td>to</td>
<td>0.00083</td>
<td>0</td>
<td>0.0017</td>
<td>0.28</td>
<td>0.00083</td>
<td>0</td>
<td>0.0025</td>
<td>0.087</td>
</tr>
<tr>
<td>eat</td>
<td>0</td>
<td>0</td>
<td>0.0027</td>
<td>0</td>
<td>0.021</td>
<td>0.0027</td>
<td>0.056</td>
<td>0</td>
</tr>
<tr>
<td>chinese</td>
<td>0.0063</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.52</td>
<td>0.0063</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>food</td>
<td>0.014</td>
<td>0</td>
<td>0.014</td>
<td>0</td>
<td>0.00092</td>
<td>0.0037</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>lunch</td>
<td>0.0059</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0029</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>spend</td>
<td>0.0036</td>
<td>0</td>
<td>0.0036</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Laplace smoothing moves too much probability mass to unseen events!
Add-$\alpha$ Smoothing

Instead of 1, add $\alpha < 1$ to each count

$$\Pr_{\alpha}(w_i|w_{i-1}) = \frac{\pi(w_{i-1}, w_i) + \alpha}{\pi(w_{i-1}) + \alpha V}$$

Choosing $\alpha$:

- Train model on training set using different values of $\alpha$
- Choose the value of $\alpha$ that minimizes cross entropy on the development set