Natural Language Processing

Info 159/259
Lecture 6: Language models 1 (Feb 4, 2021)

David Bamman, UC Berkeley
Language Model

- Vocabulary $\mathcal{V}$ is a finite set of discrete symbols (e.g., words, characters); $V = |\mathcal{V}|$

- $\mathcal{V}^+$ is the infinite set of sequences of symbols from $\mathcal{V}$; each sequence ends with STOP

- $x \in \mathcal{V}^+$
Language Model

\[ P(w) = P(w_1, \ldots, w_n) \]

\[
P(\text{"Call me Ishmael"}) = P(w_1 = \text{"call"}, w_2 = \text{"me"}, w_3 = \text{"Ishmael"}) \times P(\text{STOP})
\]

\[
\sum_{w \in V^+} P(w) = 1 \quad 0 \leq P(w) \leq 1
\]
Language Model

• Language models provide us with a way to quantify the likelihood of a sequence — i.e., plausible sentences.
To see great Pompey passe the streets of Rome:
And when you saw his Chariot but appeare,
Hauue you not made an Universall shout,
That Tyber trembled vnderneath her bankes
To heare the replication of your sounds,
Made in her Concave Shores?

- to see great Pompey passe the streets of Rome:
Machine translation

- Fidelity (to source text)
- Fluency (of the translation)
Speech Recognition

• 'Scuse me while I kiss the sky.
• 'Scuse me while I kiss this guy
• 'Scuse me while I kiss this fly.
• 'Scuse me while my biscuits fry
Dialogue generation

Q: What is your favorite animal?
A: My favorite animal is a dog.

Q: Why?
A: Because dogs are loyal and friendly.

Q: What are two reasons that a dog might be in a bad mood?
A: Two reasons that a dog might be in a bad mood are if it is hungry or if it is hot.

Q: How many bonks are in a quoit?
A: There are three bonks in a quoit.

Q: How many rainbows does it take to jump from Hawaii to seventeen?
A: It takes two rainbows to jump from Hawaii to seventeen.
Information theoretic view

“One morning I shot an elephant in my pajamas”

\[ \text{encode}(Y) \quad \text{decode(encode}(Y)) \]

\[ \text{Shannon 1948} \]
## Noisy Channel

![Table and equation]

<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASR</td>
<td>speech signal</td>
<td>transcription</td>
</tr>
<tr>
<td>MT</td>
<td>target text</td>
<td>source text</td>
</tr>
<tr>
<td>OCR</td>
<td>pixel densities</td>
<td>transcription</td>
</tr>
</tbody>
</table>

\[
P(Y \mid X) \propto P(X \mid Y) \quad P(Y) \\
\text{channel model} \quad \text{source model}
\]
Language Model

- Language modeling is the task of estimating $P(w)$
- Why is this hard?

$P(\text{"It was the best of times, it was the worst of times"})$
Chain rule (of probability)

\[ P(x_1, x_2, x_3, x_4, x_5) = P(x_1) \]
\[ \times P(x_2 \mid x_1) \]
\[ \times P(x_3 \mid x_1, x_2) \]
\[ \times P(x_4 \mid x_1, x_2, x_3) \]
\[ \times P(x_5 \mid x_1, x_2, x_3, x_4) \]
Chain rule (of probability)

\[ P(\text{"It was the best of times, it was the worst of times"}) \]
Chain rule (of probability)

\[ P(\text{"It"}) \]

\[ P(\text{"was" | "It"}) \]

\[ P(w_1) \]
\[ P(w_2 | w_1) \]
\[ P(w_3 | w_1, w_2) \]
\[ P(w_4 | w_1, w_2, w_3) \]

\[ P(w_n | w_1, \ldots, w_{n-1}) \]

P("times" | "It was the best of times, it was the worst of" )
Markov assumption

- **First-order**
  \[ P(x_i \mid x_1, \ldots x_{i-1}) \approx P(x_i \mid x_{i-1}) \]

- **Second-order**
  \[ P(x_i \mid x_1, \ldots x_{i-1}) \approx P(x_i \mid x_{i-2}, x_{i-1}) \]
Markov assumption

\[ \prod_{i}^{n} P(w_i \mid w_{i-1}) \times P(\text{STOP} \mid w_n) \]
\[ \prod_{i}^{n} P(w_i \mid w_{i-2}, w_{i-1}) \times P(\text{STOP} \mid w_{n-1}, w_n) \]
“It was the best of times, it was the worst of times”

\[
\begin{align*}
P(\text{It} \mid \text{START}_1, \text{START}_2) \\
P(\text{was} \mid \text{START}_2, \text{It}) \\
P(\text{the} \mid \text{It}, \text{was}) \\
\text{...} \\
P(\text{times} \mid \text{worst}, \text{of}) \\
P(\text{STOP} \mid \text{of}, \text{times})
\end{align*}
\]
Estimation

**unigram**

\[
\prod_{i}^{n} P(w_i) \times P(\text{STOP})
\]

\[
\prod_{i}^{n} \frac{c(w_i)}{N}
\]

**bigram**

\[
\prod_{i}^{n} P(w_i \mid w_{i-1}) \times P(\text{STOP} \mid w_n)
\]

\[
\prod_{i}^{n} \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}
\]

**trigram**

\[
\prod_{i}^{n} P(w_i \mid w_{i-2}, w_{i-1}) \times P(\text{STOP} \mid w_{n-1}, w_n)
\]

\[
\prod_{i}^{n} \frac{c(w_{i-2}, w_{i-1}, w_i)}{c(w_{i-2}, w_{i-1})}
\]
What we learn in estimating language models is $P(\text{word} \mid \text{context})$, where context — at least here — is the previous n-1 words (for ngram of order n).

We have one multinomial over the vocabulary (including STOP) for each context.
As we sample, the words we generate form the new context we condition on.

<table>
<thead>
<tr>
<th>context1</th>
<th>context2</th>
<th>generated word</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>START</td>
<td>The</td>
</tr>
<tr>
<td>START</td>
<td>The</td>
<td>dog</td>
</tr>
<tr>
<td>The</td>
<td>dog</td>
<td>walked</td>
</tr>
<tr>
<td>dog</td>
<td>walked</td>
<td>in</td>
</tr>
</tbody>
</table>
Aside: sampling?
Sampling from a Multinomial

Probability mass function (PMF)

\[ P(z = x) \] exactly
Sampling from a Multinomial

Cumulative density function (CDF)

\[ P(z \leq x) \]
Sampling from a Multinomial

Sample $p$ uniformly in $[0,1]$

Find the point $CDF^{-1}(p)$

$p = 0.78$
Sampling from a Multinomial

Sample $p$ uniformly in $[0,1]$

Find the point $\text{CDF}^{-1}(p)$

$p=.20$
Sampling from a Multinomial

Sample $p$ uniformly in $[0, 1]$

Find the point $\text{CDF}^{-1}(p)$
Unigram model

• the around, she They I blue talking “Don’t to and little come of
• on fallen used there. young people to Lázaro
• of the
• the of of never that ordered don’t avoided to complaining.
• words do had men flung killed gift the one of but thing seen I plate Bradley was by small Kingmaker.
Bigram Model

• “What the way to feel where we’re all those ancients called me one of the Council member, and smelled Tales of like a Korps peaks.”

• Tuna battle which sold or a monocle, I planned to help and distinctly.

• “I lay in the canoe ”

• She started to be able to the blundering collapsed.

• “Fine.”
Trigram Model

• “I’ll worry about it.”

• Avenue Great-Grandfather Edgeworth hasn’t gotten there.

• “If you know what. It was a photograph of seventeenth-century flourishin’ To their right hands to the fish who would not care at all. Looking at the clock, ticking away like electronic warnings about wonderfully SAT ON FIFTHDemocratic Convention in rags soaked and my past life, I managed to wring your neck a boss won’t so David Pritchet giggled.

• He humped an argument but her bare He stood next to Larry, these days it will have no trouble Jay Grayer continued to peer around the Germans weren’t going to faint in the
4gram Model

- Our visitor in an idiot sister shall be blotted out in bars and flirting with curly black hair right marble, wall-papered on screen credit.”

- You are much instant coffee ranges of hills.

- Madison might be stored here and tell everyone about was tight in her pained face was an old enemy, trading-posts of the outdoors watching Anyog extended On my lips moved feebly.

- said.

- “I’m in my mind, threw dirt in an inch,’ the Director.
Evaluation

• The best evaluation metrics are external — how does a better language model influence the application you care about?

• Speech recognition (word error rate), machine translation (BLEU score), topic models (sensemaking)
Evaluation

• A good language model should judge unseen real language to have high probability

• Perplexity = inverse probability of test data, averaged by word.

• To be reliable, the test data must be truly unseen (including knowledge of its vocabulary).

\[
\text{perplexity} = \sqrt[N]{\frac{1}{P(w_1, \ldots, w_n)}}
\]
## Experiment design

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Training Models</th>
<th>Development</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>80%</td>
<td>10%</td>
<td>10%</td>
</tr>
</tbody>
</table>

Purpose: training models; model selection; hyperparameter tuning

Evaluation: never look at it until the very end
\[ \sqrt[\sqrt{N}]{{\frac{1}{\prod_i^N P(w_i)}}} = \left( \prod_i^N P(w_i) \right)^{-\frac{1}{N}} \]

\[ = \exp \log \left( \prod_i^N P(w_i) \right)^{-\frac{1}{N}} \]

\[ = \exp \left( -\frac{1}{N} \log \prod_i^N P(w_i) \right) \]

perplexity \[ = \exp \left( -\frac{1}{N} \sum_i^N \log P(w_i) \right) \]
Perplexity

**bigram model**
(first-order markov)

\[ \text{bigram model} = \exp \left( -\frac{1}{N} \sum_{i}^{N} \log P(w_i | w_{i-1}) \right) \]

**trigram model**
(second-order markov)

\[ \text{trigram model} = \exp \left( -\frac{1}{N} \sum_{i}^{N} \log P(w_i | w_{i-2}, w_{i-1}) \right) \]
## Perplexity

<table>
<thead>
<tr>
<th>Model</th>
<th>Unigram</th>
<th>Bigram</th>
<th>Trigram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perplexity</td>
<td>962</td>
<td>170</td>
<td>109</td>
</tr>
</tbody>
</table>

SLP3 4.3
Smoothing

• When estimating a language model, we’re relying on the data we’ve observed in a training corpus.

• Training data is a small (and biased) sample of the creativity of language.
Data sparsity

<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>82</td>
<td>1</td>
<td>0</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>1</td>
<td>0</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>

Figure 4.1: Bigram counts for eight of the words (out of $V = 1446$) in the Berkeley Restaurant Project corpus of 9332 sentences. Zero counts are in gray.
As in Naive Bayes, $P(w_i) = 0$ causes $P(w) = 0$. (Perplexity?)

$$\prod_{i}^{n} P(w_i | w_{i-1}) \times P(\text{STOP} | w_n)$$
Smoothing in NB

• One solution: add a little probability mass to every element.

maximum likelihood estimate

\[ P(x_i \mid y) = \frac{n_{i,y}}{n_y} \]

smoothed estimates

If same \( \alpha \) for all \( x_i \):

\[ P(x_i \mid y) = \frac{n_{i,y} + \alpha}{n_y + V \alpha} \]

If possibly different \( \alpha \) for each \( x_i \):

\[ P(x_i \mid y) = \frac{n_{i,y} + \alpha_i}{n_y + \sum_{j=1}^{V} \alpha_j} \]
Additive smoothing

Laplace smoothing: \( \alpha = 1 \)

\[
P(w_i) = \frac{c(w_i) + \alpha}{N + V\alpha}
\]

\[
P(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i) + \alpha}{c(w_{i-1}) + V\alpha}
\]
Smoothing

MLE

Smoothing is the re-allocation of probability mass

smoothing with $\alpha = 1$
Smoothing

• How can best re-allocate probability mass?

Interpolation

• As ngram order rises, we have the potential for higher precision but also higher variability in our estimates.

• A linear interpolation of any two language models p and q (with $\lambda \in [0,1]$) is also a valid language model.

$$\lambda p + (1 - \lambda)q$$

p = the web            q = political speeches
Interpolation

- We can use this fact to make higher-order language models more robust.

\[ P(w_i \mid w_{i-2}, w_{i-1}) = \lambda_1 P(w_i \mid w_{i-2}, w_{i-1}) \]
\[ + \lambda_2 P(w_i \mid w_{i-1}) \]
\[ + \lambda_3 P(w_i) \]

\[ \lambda_1 + \lambda_2 + \lambda_3 = 1 \]
Interpolation

• How do we pick the best values of $\lambda$?
  
  • Grid search over development corpus

  • Expectation-Maximization algorithm (treat as missing parameters to be estimated to maximize the probability of the data we see).
Kneser-Ney smoothing

• Intuition: When backing off to a lower-order ngram, maybe the overall ngram frequency is not our best guess.

  I can’t see without my reading ____________

  \[ P(“Francisco”) \; > \; P(“glasses”) \]

• *Francisco* is more frequent, but shows up in fewer unique bigrams ("San Francisco") — so we shouldn’t expect it in new contexts; *glasses*, however, does show up in many different bigrams
Kneser-Ney smoothing

• Intuition: estimate how likely a word is to show up in a new continuation?

• How many different bigram types does a word type $w$ show up in (normalized by all bigram types that are seen)

\[
\frac{|v \in \mathcal{V} : c(v, w) > 0|}{|v', w' \in \mathcal{V} : c(v', w') > 0|}
\]
\( P_{\text{CONTINUATION}}(w) = \frac{|v \in \mathcal{V} : c(v, w) > 0|}{|v', w' \in \mathcal{V} : c(v', w') > 0|} \)

\( P_{\text{CONTINUATION}}(\text{Francisco}) \approx \frac{149}{10000000} \)

\( P_{\text{CONTINUATION}}(\text{dog}) \approx \frac{1391}{10000000} \)

\( P_{\text{CONTINUATION}}(w) \) is the continuation probability for the unigram \( w \) (the frequency with which it appears as the suffix in distinct bigram types)
Kneser-Ney smoothing

\[
\max\left\{\frac{c(w_{i-1}, w_i) - d, 0}{c(w_{i-1})}\right\} + \lambda(w_{i-1}) P_{CONTINUATION}(w_i)
\]
Kneser-Ney smoothing

\[
\frac{\max\{c(w_{i-1}, w_i) - d, 0\}}{c(w_{i-1})}
\]

d is a discount factor (usually between 0 and 1 — how much we discount the observed counts by)

discounted bigram probability
Kneser-Ney smoothing

\[
\lambda(w_{i-1}) = d \frac{|w_i \in \mathcal{V} : c(w_{i-1}, w_i) > 0|}{c(w_{i-1})}
\]

\(\lambda\) here captures the discounted mass we’re reallocating from prefix \(w_{i-1}\)
Kneser-Ney smoothing

\[ \lambda(\text{red}) = 1 \times \frac{3}{15} \]

<table>
<thead>
<tr>
<th>(w_{i-1})</th>
<th>(w_i)</th>
<th>(C(w_{i-1}, w_i))</th>
<th>(C(w_{i-1}, w_i) - d(1))</th>
</tr>
</thead>
<tbody>
<tr>
<td>red</td>
<td>hook</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>red</td>
<td>car</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>red</td>
<td>watch</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>sum</td>
<td></td>
<td>15</td>
<td>12</td>
</tr>
</tbody>
</table>

12/15 of the probability mass stays with the original counts; 3/15 is reallocated.
\[ P_{\text{CONTINUATION}}(w) = \frac{|v \in \mathcal{V} : c(v, w) > 0|}{|v', w' \in \mathcal{V} : c(v', w') > 0|} \]

\[ \max \left\{ \frac{c(w_{i-1}, w_i) - d,0}{c(w_{i-1})} + \lambda(w_{i-1}) P_{\text{CONTINUATION}}(w_i) \right\} \]
we'll move all of the mass we subtracted here over to this side

\[
\max\left\{\frac{c(w_{i-1}, w_i) - d, 0}{c(w_{i-1})}\right\} + \lambda(w_{i-1}) P_{CONTINUATION}(w_i)
\]

and distribute it according to the continuation probability
“Stupid backoff”

\[
S(w_i \mid w_{i-k+1}, \ldots, w_{i-1}) = \begin{cases} 
\frac{c(w_{i-k+1}, \ldots, w_i)}{c(w_{i-k+1}, \ldots, w_{i-1})} & \text{if full sequence observed} \\
= \lambda S(w_i \mid w_{i-k+2}, \ldots, w_{i-1}) & \text{otherwise}
\end{cases}
\]

No discounting here, just back off to lower order ngram if the higher order is not observed.

Cheap to calculate; works almost as well as KN when there is a lot of data

Why?

• Language models give us an estimate for the probability of a sequence, which is directly useful for applications that are deciding between different sentences) as viable outputs:
  • Machine translation
  • Speech recognition
  • OCR
  • Dialogue agents
Why?

• Language models directly allow us to predict the next word in a sequence (useful for autocomplete).
Why?

• Language models can directly encode knowledge present in the training corpus.

The director of *2001: A Space Odyssey* is ___________
Why?

- Language models can directly encode knowledge present in the training corpus.

<table>
<thead>
<tr>
<th>Query</th>
<th>Answer</th>
<th>Generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Francesco Bartolomeo Conti was born in ____</td>
<td>Florence</td>
<td>Rome [-1.8], Florence [-1.8], Naples</td>
</tr>
<tr>
<td>Adolphe Adam died in ____</td>
<td>Paris</td>
<td>Paris [-0.5], London [-3.5], Vienna</td>
</tr>
<tr>
<td>English bulldog is a subclass of ____</td>
<td>dog</td>
<td>dogs [-0.3], breeds [-2.2], dog</td>
</tr>
<tr>
<td>The official language of Mauritius is ____</td>
<td>English</td>
<td>English [-0.6], French [-0.9], Arabic</td>
</tr>
<tr>
<td>Patrick Oboya plays in ____ position.</td>
<td>midfielder</td>
<td>centre [-2.0], center [-2.2], midfielder</td>
</tr>
<tr>
<td>Hamburg Airport is named after ____</td>
<td>Hamburg</td>
<td>Hess [-7.0], Hermann [-7.1], Schmidt</td>
</tr>
</tbody>
</table>

Why?

- Language modeling turns out to be a good proxy task for learning about linguistic structure.
- See contextual word embeddings (BERT/ELMo), in class 2/16.
You should feel comfortable:

• Calculate the probability of a sentence given a trained model
• Estimating (e.g., trigram) language model
• Evaluating perplexity on held-out data
• Sampling a sentence from a trained model