Language Models

Prof. Srijan Kumar
with Arindum Roy and Roshan Pati
Today’s Lecture

• What are language models?
  - Statistical language models
  • Neural language models
Language Models: Objective

- Key question: How well does a model represent the language?
  - **Character language model**: Given alphabet vocabulary $V$, models the probability of generating strings in the language.
  - **Word language model**: Given word vocabulary $V$, models the probability of generating sentences in the language.
Language Model: Applications

• **Assign a probability to sentences**
  – **Machine translation:**
    • $P(\text{high wind tonight}) > P(\text{large wind tonight})$
  – **Spell correction:**
    • The office is about fifteen *minuets* from my house
    • $P(\text{about fifteen minutes from}) > P(\text{about fifteen minuets from})$
  – **Speech recognition:**
    • $P(\text{I saw a van}) >> P(\text{eyes awe of an})$
  – **Information retrieval:** use words that you expect to find in matching documents as your query
  – Many more: Summarization, question-answering, and more
Today’s Lecture

- What are language models?
  - Statistical language models
  - Neural language models
Language Model: Definition

- **Goal:** compute the probability of a sentence or sequence of words: \( P(S) = P(w_1, w_2, \ldots w_n) \)

- **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 is a language model

- **How to compute the joint probability?**
  - Intuition: apply the chain rule
Types of Language Models

• The types of language models depend on the complexity on the word-word or character-character dependency they can handle

• Common types are:
  – Unigram language model
  – Bigram language model
  – N-gram language model
How To Compute Sentence Probability?

- Given sentence $S = t_1t_2t_3t_4$
- Applying the chain rule under language model $M$

$$P(t_1t_2t_3t_4 | M) = P(t_1 | M)P(t_2 | M, t_1)P(t_3 | M, t_1t_2)P(t_4 | M, t_1t_2t_3)$$
Unigram Model

- Unigram language model only models the probability of each word according to the model
  - Does NOT model word-word dependency
  - The word order is irrelevant
  - Akin to the “bag of words” model

\[ P(t_1 t_2 t_3 t_4 | M) = P(t_1 | M) P(t_2 | M) P(t_3 | M) P(t_4 | M) \]
Bigram Model

- Bigram language model models the consecutive word dependency
  - Does NOT model longer dependency
  - Word order is relevant here

\[
P(t_1t_2t_3t_4|\mathcal{M}) = P(t_1|\mathcal{M})P(t_2|\mathcal{M}, t_1)P(t_3|\mathcal{M}, t_2)P(t_4|\mathcal{M}, t_3)
\]
N-gram Model

- Bigram language model models the longer sequences of word dependency
  - Most complex among all three

\[ P(t_1 t_2 t_3 t_4 | M) = P(t_1 | M) P(t_2 | M, t_1) P(t_3 | M, t_1 t_2) P(t_4 | M, t_1 t_2 t_3) \]
Unigram Language Model: Example

- What is the probability of the sentence $s$ under language model $M$?
- Example:

  “the man likes the woman”
  \[
  0.2 \times 0.01 \times 0.02 \times 0.2 \times 0.01 = 0.00000008
  \]

  $P(s \mid M) = 0.00000008$

<table>
<thead>
<tr>
<th>Word</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>0.2</td>
</tr>
<tr>
<td>a</td>
<td>0.1</td>
</tr>
<tr>
<td>man</td>
<td>0.01</td>
</tr>
<tr>
<td>woman</td>
<td>0.01</td>
</tr>
<tr>
<td>said</td>
<td>0.03</td>
</tr>
<tr>
<td>likes</td>
<td>0.02</td>
</tr>
</tbody>
</table>
Comparing Language Models

• **Given two language models, how can we decide which language model is better?**

• **Solution:**
  – Take a set $S$ of sentences we desire to model
  – For each language model:
    • Find the probability of each sentence
    • Average the probability scores
  – The language model with the highest average probability is the best fit for language model
Comparing Language Models

• s: “the man likes the woman”
• M1: \(0.2 \times 0.01 \times 0.02 \times 0.2 \times 0.01\) \(\Rightarrow P(s|M1) = 0.00000008\)
• M2: \(0.1 \times 0.1 \times 0.01 \times 0.1 \times 0.1\) \(\Rightarrow P(s|M2) = 0.000001\)
• \(P(s|M2) > P(s|M1) \Rightarrow M2\) is a better language model

<table>
<thead>
<tr>
<th>Word</th>
<th>Probability</th>
<th>Word</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>0.2</td>
<td>the</td>
<td>0.1</td>
</tr>
<tr>
<td>a</td>
<td>0.1</td>
<td>a</td>
<td>0.02</td>
</tr>
<tr>
<td>man</td>
<td>0.01</td>
<td>man</td>
<td>0.1</td>
</tr>
<tr>
<td>woman</td>
<td>0.01</td>
<td>woman</td>
<td>0.1</td>
</tr>
<tr>
<td>said</td>
<td>0.03</td>
<td>said</td>
<td>0.02</td>
</tr>
<tr>
<td>likes</td>
<td>0.02</td>
<td>likes</td>
<td>0.01</td>
</tr>
</tbody>
</table>
Estimating Probabilities

- N-gram conditional probability can be estimated based on the **raw occurrence counts in the observed corpus**
- Uni-gram
  \[ P(t_n) = C(t_n) \]
- Bi-gram
  \[ P(t_n|t_{n-1}) = \frac{C(t_{n-1}t_n)}{C(t_{n-1})} \]
- N-gram
  \[ P(t_n|t_{n-N-1:n-1}) = \frac{C(t_{n-N-1} \ldots t_{n-1}t_n)}{C(t_{n-N-1} \ldots t_{n-1})} \]
Estimating Bigram Probabilities

• Corpus: Berkeley Restaurant Project sentences
  – 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 cafe venezia open during the day
### Raw Bigram Counts

- **Bigram matrix created from 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>
# Raw Bigram Probabilities

- **Unigram 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>count</td>
<td>2533</td>
<td>927</td>
<td>2417</td>
<td>746</td>
<td>158</td>
<td>1093</td>
<td>341</td>
<td>278</td>
</tr>
</tbody>
</table>

- **Normalize by unigrams**

\[
P(want \mid i) = \frac{C(i, want)}{C(i)} = \frac{827}{2533} = 0.33\]
Next Lecture

• What are language models?
  • Statistical language models
  • Neural language models