NLP Programming Tutorial 7 - Topic Models

Graham Neubig
Nara Institute of Science and Technology (NAIST)
Topics in Documents

- In general, documents can be grouped into topics

<table>
<thead>
<tr>
<th>Cuomo to Push for Broader Ban on Assault Weapons</th>
<th>2012 Was Hottest Year in U.S. History</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Topics in Documents

- In general, documents can be grouped into topics

```
Cuomo to Push for Broader Ban on Assault Weapons

...  ...

New York  Weather
Politics  Climate
Weapons   Statistics
Crime     U.S.
```

2012 Was Hottest Year in U.S. History

...  ...

...  ...

...  ...
Topic Modeling

- Topic modeling finds topics $Y$ given documents $X$

- A type of “structured” prediction
Probabilistic Generative Model

- We assume some probabilistic model generated the topics $Y$ and documents $X$ jointly

$$P(Y, X)$$

- The topics $Y$ with highest joint probability given $X$ also has the highest conditional probability

$$\arg\max_Y P(Y|X) = \arg\max_Y P(Y, X)$$
Generative Topic Model

- Assume we have words $X$ and topics $Y$:

  $X = \text{Cuomo to Push for Broader Ban on Assault Weapons}$

  $Y = \text{NY Func Pol Func Pol Pol Func Crime Crime}$

  NY=New York, Func=Function Word, Pol=Politics, Crime=Crime

- First decide topics (independently)

  \[ P(Y) = \prod_{i=1}^{I} P(y_i) \]

- Then decide words given topics (independently)

  \[ P(X|Y) = \prod_{i=1}^{I} P(x_i|y_i) \]
Unsupervised Topic Modeling

- Given only the documents \( X \), find topic-like clusters \( Y \)

<table>
<thead>
<tr>
<th>( X )</th>
<th>( Y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cuomo to Push for Broader Ban on Assault Weapons</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
</tr>
<tr>
<td>24</td>
<td>18</td>
</tr>
<tr>
<td>10</td>
<td>49</td>
</tr>
<tr>
<td>19</td>
<td>37</td>
</tr>
</tbody>
</table>

- A type of “structured” prediction
- But unlike before, we have no labeled training data!
Latent Dirichlet Allocation

- Most popular generative model for topic modeling
- First generate model parameters $\theta$: $P(\theta)$
- For every document in $X$:
  - Generate document topic distribution $T_i$: $P(T_i|\theta)$
  - For each word $x_{i,j}$ in $X_i$:
    - Generate word topic $y_{i,j}$: $P(y_{i,j}|T_i)$
    - Generate the word $x_{i,j}$: $P(x_{i,j}|y_{i,j},\theta)$

$$P(X,Y) = \int_\theta P(\theta) \prod_i P(T_i|\theta) \prod_j P(y_{i,j}|T_i,\theta) P(x_{i,j}|y_{i,j},\theta)$$
Maximum Likelihood Estimation

- Assume we have words $X$ and topics $Y$:

<table>
<thead>
<tr>
<th>$X_1$ = Cuomo to Push for Broader Ban on Assault Weapons</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Y_1 = 32 \ 7 \ 24 \ 7 \ 24 \ 24 \ 7 \ 10 \ 10$</td>
</tr>
</tbody>
</table>

- Can decide the topic distribution for each document:

$$P(y | Y_i) = \frac{c(y, Y_i)}{|Y_i|} \quad \text{e.g.: } P(y = 24 | Y_1) = \frac{3}{9}$$

- Can decide word distribution for each topic:

$$P(x | y) = \frac{c(x, y)}{c(y)} \quad \text{e.g.: } P(x = \text{assault} | y = 10) = \frac{1}{2}$$
Problem: Unobserved Variables

- **Problem**: We do not know the values of $y_{i,j}$
- **Solution**: Use a method for unsupervised learning
  - EM Algorithm
  - Variational Bayes
  - **Sampling**
Sampling Basics

- Generate a sample from probability distribution:

  Distribution: $P(\text{Noun})=0.5\quad P(\text{Verb})=0.3\quad P(\text{Preposition})=0.2$


- Count the samples and calculate probabilities

  $P(\text{Noun})= 4/10 = 0.4$, $P(\text{Verb})= 4/10 = 0.4$, $P(\text{Preposition}) = 2/10 = 0.2$

- More samples = better approximation
Actual Algorithm

\[
\text{SampleOne}(\text{probs})
\]

\[
z = \text{Sum}(\text{probs})
\]

\[
\text{remaining} = \text{Rand}(z)
\]

\[
\text{for each } i \text{ in } 0 \ldots \text{probs.size}-1
\]

\[
\text{remaining} -= \text{probs}[i]
\]

\[
\text{if } \text{remaining} <= 0
\]

\[
\text{return } i
\]

Bug check, beware of overflow!

Calculate sum of probs

Generate number from uniform distribution over \([0,z)\)

Iterate over all probabilities

Subtract current prob. value

If smaller than zero, return current index as answer
Gibbs Sampling

- Want to sample a 2-variable distribution $P(A,B)$
  - ... but cannot sample directly from $P(A,B)$
  - ... but can sample from $P(A|B)$ and $P(B|A)$
- **Gibbs sampling** samples variables one-by-one to recover true distribution
- Each iteration:
  - Leave $A$ fixed, sample $B$ from $P(B|A)$
  - Leave $B$ fixed, sample $A$ from $P(A|B)$
Example of Gibbs Sampling

- **Parent A** and **child B** are shopping, what sex?
  
  \[
  \begin{align*}
  P(\text{Mother} | \text{Daughter}) &= \frac{5}{6} = 0.833 \\
  P(\text{Mother} | \text{Son}) &= \frac{5}{8} = 0.625 \\
  P(\text{Daughter} | \text{Mother}) &= \frac{2}{3} = 0.667 \\
  P(\text{Daughter} | \text{Father}) &= \frac{2}{5} = 0.4
  \end{align*}
  \]

- **Original state:** Mother/Daughter
  
  Sample \( P(\text{Mother} | \text{Daughter}) = 0.833 \), chose Mother
  
  Sample \( P(\text{Daughter} | \text{Mother}) = 0.667 \), chose Son
  
  \( c(\text{Mother, Son})++ \)
  
  Sample \( P(\text{Mother} | \text{Son}) = 0.625 \), chose Mother
  
  Sample \( P(\text{Daughter} | \text{Mother}) = 0.667 \), chose Daughter
  
  \( c(\text{Mother, Daughter})++ \)

  ...
Try it Out:

- In this case, we can confirm this result by hand
Sampling in Topic Models (1)

- Sample one $y_{i,j}$ at a time:

$$X_1 = \text{Cuomo to Push for Broader Ban on Assault Weapons}$$

$$Y_1 = \begin{pmatrix} 5 & 7 & 4 & 7 & 3 & 4 & 7 & 6 & 6 \end{pmatrix}$$

- Subtract of $y_{i,j}$ and re-calculate topics and parameters

$$\{0, 0, \frac{1}{9}, \frac{2}{9}, \frac{1}{9}, \frac{2}{9}, \frac{3}{9}, 0\}$$

$$\{0, 0, \frac{1}{8}, \frac{2}{8}, \frac{1}{8}, \frac{2}{8}, \frac{2}{8}, 0\}$$
Sampling in Topic Models (2)

- Sample one $y_{i,j}$ at a time:

\[X_1 = \text{Cuomo to Push for Broader Ban on Assault Weapons}\]
\[Y_1 = 5 \quad 7 \quad 4 \quad ???? \quad 3 \quad 4 \quad 7 \quad 6 \quad 6\]

- Multiply topic prob., by word given topic prob.:

\[
P(y_{i,j} | T_i) = \{ 0, 0, 0.125, 0.25, 0.125, 0.25, 0.25, 0 \}
\]

\[
P(x_{i,j} | y_{i,j}, \theta) = \{0.01, 0.02, 0.01, 0.10, 0.08, 0.07, 0.70, 0.01\}
\]

\[
P(x_{i,j} | y_{i,j}, T_i, \theta) = \{0, 0, 0.00125, 0.01, 0.01, 0.00875, 0.175, 0\}/Z
\]

Calculated from whole corpus

Normalization constant
Sampling in Topic Models (3)

- Sample one value from this distribution:

\[
P(x_{i,j}, y_{i,j} | T_i, \theta) = \{0, 0, 0.00125, 0.01, 0.01, 0.00875, 0.175, 0\}/Z
\]

- Add the word with the new topic:

\[
X_1 = \text{Cuomo to Push for Broader Ban on Assault Weapons}
\]

\[
Y_1 = 5 \quad 7 \quad 4 \quad 6 \quad 3 \quad 4 \quad 7 \quad 6 \quad 6
\]

- Update the counts and the probabilities:

\[
\{0, 0, 1/8, 2/8, 1/8, 2/8, 2/8, 0\}
\]

\[
\{0, 0, 1/9, 2/9, 1/9, 3/9, 2/9, 0\}
\]
Dirichlet Smoothing

- **Problem:** Many probabilities are zero!
  - Cannot escape from local minima

- **Solution:** Smooth the probabilities

\[
P(x_{i,j}|x_{i,j}) = \frac{c(x_{i,j}, y_{i,j})}{c(y_{i,j})} \quad \rightarrow \quad P(x_{i,j}|y_{i,j}) = \frac{c(x_{i,j}, y_{i,j}) + \alpha}{c(y_{i,j}) + \alpha * N_x}
\]

\[
P(y_{i,j}|y_i) = \frac{c(y_{i,j}, Y_i)}{c(Y_i)} \quad \rightarrow \quad P(y_{i,j}|Y_i) = \frac{c(y_{i,j}|Y_i) + \beta}{c(Y_i) + \beta * N_y}
\]

- \(N_x\) and \(N_y\) are number of unique words and topics
- Equal to using a Dirichlet prior over the probabilities (More details in my Bayes tutorial)
Implementation: Initialization

make vectors xcorpus, ycorpus # to store each value of x, y
make map xcounts, ycounts # to store counts for probs
for line in file
    docid = size of xcorpus # get a numerical ID for this doc
    split line into words
    make vector topics # create random topic ids
    for word in words
        topic = RAND(NUM_TOPICS) # random in [0,NUM_TOP)
        append topic to topics
        ADD_COUNTS(word, topic, docid, 1) # add counts
    append words (vector) to xcorpus
    append topics (vector) to ycorpus
Implementation: Adding Counts

**ADD_COUNTS** *(word, topic, docid, amount)*

- \( x\text{counts}[\text{topic}] += \text{amount} \)
- \( x\text{counts}[\text{word,topic}] += \text{amount} \)
- \( y\text{counts}[\text{docid}] += \text{amount} \)
- \( y\text{counts}[\text{topic,docid}] += \text{amount} \)

**bug check!**

if any of these values < 0, throw error
Implementation: Sampling

```python
for many iterations:
    ll = 0
    for i in range(len(xcorpus)):
        for j in range(len(xcorpus[i])):
            x = xcorpus[i][j]
            y = ycorpus[i][j]
            AddCounts(x, y, i, -1)  # subtract the counts (hence -1)
            make vector probs
            for k in range(NUM_TOPICS-1):
                append P(x|k) * P(k|Y) to probs  # prob of topic k
            new_y = SAMPLEONE(probs)
            ll += log(probs[new_y])  # Calculate the log likelihood
            AddCounts(x, new_y, i, 1)  # add the counts
            ycorpus[i][j] = new_y
print ll
print out wcounts and tcounts
```
Exercise
Exercise

- **Write** `learn-lda`
- **Test** the program, setting `NUM_TOPICS` to 2
  - **Input:** `test/07-train.txt`
  - **Answer:**
    - No correct answer! (Because sampling is random)
    - However, “a b c d” and “e f g h” should probably be different topics
- **Train** a topic model on `data/wiki-en-documents.word` with 20 topics
- **Find** some topics that match with your intuition
- **Challenge:** Change the model so you don't have to choose the number of topics in advance (Read about non-parametric Bayesian techniques)
Thank You!