

## Supplementary: Topic Modeling

65070501037

Paweeorn Soratyathorn

Objectives:

- To demonstrate students how to apply topic modeling to real-world data.
- Students will gain hands-on experience through this example.

Create a data directory and store the downloaded data (state-of-the-union.csv) in it. The data is about State of the Union addresses from 1970 to 2012.

```
1 !mkdir -p data
2 !wget -nc https://nyc3.digitaloceanspaces.com/ml-files-distro/v1/text-analysis/data/state-of-the-union.csv -P data
```

--2024-11-05 16:42:39-- https://nyc3.digitaloceanspaces.com/ml-files-distro/v1/text-analysis/data/state-of-the-union.csv  
Resolving nyc3.digitaloceanspaces.com (nyc3.digitaloceanspaces.com)... 162.243.189.2  
Connecting to nyc3.digitaloceanspaces.com (nyc3.digitaloceanspaces.com)|162.243.189.2|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 10501219 (10M) [text/csv]  
Saving to: 'data/state-of-the-union.csv'

state-of-the-union. 100%[=====] 10.01M 55.0MB/s in 0.2s

2024-11-05 16:42:39 (55.0 MB/s) - 'data/state-of-the-union.csv' saved [10501219/10501219]

Read data

```
1 import pandas as pd
2 import numpy as np
3
4 df = pd.read_csv("data/state-of-the-union.csv")
5
6 # Clean it up a little bit, removing non-word characters (numbers and __ etc)
7 df.content = df.content.str.replace("[^A-Za-z ]", " ")
8
9 df.head()
```

	year	content
0	1790	George Washington\nJanuary 8, 1790\nFellow-C...
1	1790	\nState of the Union Address\nGeorge Washingto...
2	1791	\nState of the Union Address\nGeorge Washingto...
3	1792	\nState of the Union Address\nGeorge Washingto...
4	1793	\nState of the Union Address\nGeorae Washingato...

Next steps: [Generate code with df](#) [View recommended plots](#) [New interactive sheet](#)

1 df.shape

(226, 2)


Using Gensim to perform topic modeling

```
1 # Run this cell if gensim has not been installed yet.
2 !pip install gensim
```

Requirement already satisfied: gensim in /usr/local/lib/python3.10/dist-packages (4.3.3)  
Requirement already satisfied: numpy<2.0,>=1.18.5 in /usr/local/lib/python3.10/dist-packages (from gensim) (1.26.4)  
Requirement already satisfied: scipy<1.14.0,>=1.7.0 in /usr/local/lib/python3.10/dist-packages (from gensim) (1.13.1)  
Requirement already satisfied: smart-open>=1.8.1 in /usr/local/lib/python3.10/dist-packages (from gensim) (7.0.5)  
Requirement already satisfied: wrapt in /usr/local/lib/python3.10/dist-packages (from smart-open>=1.8.1->gensim) (1.16.0)


Apply `simple_process` to convert a document into a list of tokens. The input will be lowercased, tokenized, and de-accented (optional).

```
1 from gensim.utils import simple_preprocess
2
3 texts = df.content.apply(simple_preprocess)
4 texts
```



	content
0	[george, washington, january, fellow, citizens...
1	[state, of, the, union, address, george, washi...
2	[state, of, the, union, address, george, washi...
3	[state, of, the, union, address, george, washi...
4	[state, of, the, union, address, george, washi...
...	...
221	[state, of, the, union, address, george, bush,...
222	[address, to, joint, session, of, congress, ba...
223	[state, of, the, union, address, barack, obama...
224	[state, of, the, union, address, barack, obama...
225	[state, of, the, union, address, barack, obama...

226 rows × 1 columns




Create a dictionary, using the texts that have already been preprocessed.

The method `doc2bow` is for converting document (a list of words) into the bag-of-words format.


```
1 from gensim import corpora
2
3 dictionary = corpora.Dictionary(texts)
4 dictionary.filter_extremes(no_below=5, no_above=0.5, keep_n=2000)
5 corpus = [dictionary.doc2bow(text) for text in texts]
```

Specify number of topics manually. Let's try 2 topics first.

```
1 from gensim import models
2
3 n_topics = 2
4
5 lda_model = models.LdaModel(corpus=corpus, num_topics=n_topics)
6 lda_model.print_topics()
```




```
WARNING:gensim.models.ldamodel:no word id mapping provided; initializing from corpus, assuming identity
WARNING:gensim.models.ldamodel:too few updates, training might not converge; consider increasing the number of passes or iterations to impr
[(0,
  '0.003*"1930" + 0.003*"1986" + 0.003*"1260" + 0.003*"1559" + 0.002*"1971" + 0.002*"1626" + 0.002*"1784" + 0.002*"1242" + 0.002*"440" +
  0.002*"1922"'),
 (1,
  '0.004*"1559" + 0.003*"1260" + 0.003*"1930" + 0.003*"1986" + 0.002*"1242" + 0.002*"151" + 0.002*"1971" + 0.002*"1626" + 0.002*"578" +
  0.002*"1989"')]
```




Let's try 5 topics.

```
1 n_topics = 5
2
3 lda_model = models.LdaModel(corpus=corpus, num_topics=n_topics)
4 lda_model.print_topics()
```



```
WARNING:gensim.models.ldamodel:no word id mapping provided; initializing from corpus, assuming identity
WARNING:gensim.models.ldamodel:too few updates, training might not converge; consider increasing the number of passes or iterations to impr
[(0,
  '0.004*"1260" + 0.004*"1986" + 0.003*"1964" + 0.003*"1559" + 0.003*"1930" + 0.003*"1974" + 0.003*"1242" + 0.003*"1971" + 0.003*"1989" +
  0.002*"1999"'),
 (1,
  '0.005*"1930" + 0.004*"1971" + 0.003*"1260" + 0.003*"1986" + 0.003*"1559" + 0.003*"1999" + 0.002*"1242" + 0.002*"1802" + 0.002*"1964" +
  0.002*"1995"'),
 (2,
  '0.004*"1260" + 0.003*"1930" + 0.003*"1559" + 0.003*"1986" + 0.002*"1971" + 0.002*"1626" + 0.002*"151" + 0.002*"1242" + 0.002*"1999" +
  0.002*"1327"'),
 (3,
  '0.004*"1559" + 0.003*"1626" + 0.003*"1986" + 0.003*"1930" + 0.002*"951" + 0.002*"1260" + 0.002*"151" + 0.002*"1865" + 0.002*"1242" +
  0.002*"1446"'),
 (4,
  '0.003*"1559" + 0.003*"151" + 0.003*"1784" + 0.002*"1986" + 0.002*"976" + 0.002*"1446" + 0.002*"1327" + 0.002*"1242" + 0.002*"62" +
  0.002*"57"')]
```



Let's try 15 topics.

```

1 n_topics = 15
2
3 lda_model = models.LdaModel(corpus=corpus, num_topics=n_topics)
4 lda_model.print_topics()

```

WARNING:gensim.models.ldamodel:no word id mapping provided; initializing from corpus, assuming identity  
 WARNING:gensim.models.ldamodel:too few updates, training might not converge; consider increasing the number of passes or iterations to impr

```

[0,
  '0.004*"1986" + 0.004*"1930" + 0.004*"1260" + 0.003*"1559" + 0.003*"1242" + 0.003*"1784" + 0.002*"1974" + 0.002*"1626" + 0.002*"1971" +
  0.002*"266"'),
  (1,
    '0.004*"1559" + 0.003*"440" + 0.002*"151" + 0.002*"976" + 0.002*"1260" + 0.002*"468" + 0.002*"1242" + 0.002*"1986" + 0.002*"1974" +
    0.002*"1971"'),
  (2,
    '0.004*"1930" + 0.004*"1242" + 0.003*"1986" + 0.003*"1260" + 0.003*"1999" + 0.003*"1964" + 0.003*"1974" + 0.003*"1989" + 0.003*"1922" +
    0.002*"1626"'),
  (3,
    '0.004*"1260" + 0.004*"1971" + 0.004*"1930" + 0.003*"1986" + 0.003*"1999" + 0.003*"1995" + 0.003*"1242" + 0.002*"1989" + 0.002*"1964" +
    0.002*"1559"'),
  (4,
    '0.006*"1559" + 0.003*"976" + 0.003*"1619" + 0.003*"1626" + 0.003*"1784" + 0.002*"951" + 0.002*"1792" + 0.002*"62" + 0.002*"578" +
    0.002*"1017"'),
  (5,
    '0.004*"1559" + 0.003*"1986" + 0.003*"1327" + 0.002*"1446" + 0.002*"1922" + 0.002*"1626" + 0.002*"1971" + 0.002*"1260" + 0.002*"578" +
    0.002*"151"'),
  (6,
    '0.005*"1260" + 0.003*"1559" + 0.003*"1930" + 0.003*"151" + 0.003*"1626" + 0.003*"1995" + 0.002*"1971" + 0.002*"1327" + 0.002*"1986" +
    0.002*"1651"'),
  (7,
    '0.003*"1986" + 0.003*"1971" + 0.003*"1242" + 0.003*"1559" + 0.003*"151" + 0.002*"1626" + 0.002*"1930" + 0.002*"1260" + 0.002*"1989" +
    0.002*"1784"'),
  (8,
    '0.003*"1559" + 0.003*"1260" + 0.003*"19" + 0.002*"1446" + 0.002*"1930" + 0.002*"266" + 0.002*"440" + 0.002*"57" + 0.002*"1784" +
    0.002*"62"'),
  (9,
    '0.006*"1559" + 0.004*"1930" + 0.003*"1260" + 0.003*"151" + 0.003*"951" + 0.002*"1999" + 0.002*"266" + 0.002*"57" + 0.002*"1242" +
    0.002*"1986"'),
  (10,
    '0.004*"1930" + 0.003*"1986" + 0.003*"1242" + 0.003*"1971" + 0.003*"1974" + 0.002*"1260" + 0.002*"1989" + 0.002*"1995" + 0.002*"1697" +
    0.002*"1626"'),
  (11,
    '0.004*"1986" + 0.003*"1626" + 0.002*"1559" + 0.002*"1930" + 0.002*"1865" + 0.002*"976" + 0.002*"440" + 0.002*"1545" + 0.002*"1260" +
    0.002*"1242"'),
  (12,
    '0.004*"1930" + 0.004*"1260" + 0.003*"1971" + 0.003*"1986" + 0.003*"1922" + 0.002*"1784" + 0.002*"1242" + 0.002*"1327" + 0.002*"1545" +
    0.002*"1989"'),
  (13,
    '0.003*"1930" + 0.003*"1986" + 0.003*"951" + 0.002*"1260" + 0.002*"1446" + 0.002*"1559" + 0.002*"1242" + 0.002*"266" + 0.002*"1971" +
    0.002*"62"'),
  (14,
    '0.005*"1559" + 0.003*"1930" + 0.003*"1986" + 0.003*"1626" + 0.003*"1865" + 0.002*"1964" + 0.002*"1260" + 0.002*"151" + 0.002*"92" +
    0.002*"57"')]

```

```

1 # Run this cell if pyLDAvis has never been installed
2 !pip install pyLDAvis

```


Collecting pyLDAvis  
 Downloading pyLDAvis-3.4.1-py3-none-any.whl.metadata (4.2 kB)  
 Requirement already satisfied: numpy>=1.24.2 in /usr/local/lib/python3.10/dist-packages (from pyLDAvis) (1.26.4)  
 Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from pyLDAvis) (1.13.1)  
 Requirement already satisfied: pandas>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from pyLDAvis) (2.2.2)  
 Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from pyLDAvis) (1.4.2)  
 Requirement already satisfied: Jinja2 in /usr/local/lib/python3.10/dist-packages (from pyLDAvis) (3.1.4)  
 Requirement already satisfied: Numexpr in /usr/local/lib/python3.10/dist-packages (from pyLDAvis) (2.10.1)  
 Collecting funcy (from pyLDAvis)  
 Downloading funcy-2.0-py2.py3-none-any.whl.metadata (5.9 kB)  
 Requirement already satisfied: scikit-learn>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from pyLDAvis) (1.5.2)  
 Requirement already satisfied: gensim in /usr/local/lib/python3.10/dist-packages (from pyLDAvis) (4.3.3)  
 Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from pyLDAvis) (75.1.0)  
 Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas>=2.0.0->pyLDAvis) (2.8.2)  
 Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=2.0.0->pyLDAvis) (2024.2)  
 Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas>=2.0.0->pyLDAvis) (2024.2)  
 Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn>=1.0.0->pyLDAvis) (3.5.0)  
 Requirement already satisfied: smart-open>=1.8.1 in /usr/local/lib/python3.10/dist-packages (from gensim->pyLDAvis) (7.0.5)  
 Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from Jinja2->pyLDAvis) (3.0.2)  
 Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas>=2.0.0->pyLDAvis) (1.16.0)  
 Requirement already satisfied: wrapt in /usr/local/lib/python3.10/dist-packages (from smart-open>=1.8.1->gensim->pyLDAvis) (1.16.0)  
 Downloading pyLDAvis-3.4.1-py3-none-any.whl (2.6 MB)  
 2.6/2.6 MB 25.2 MB/s eta 0:00:00  
 Downloading funcy-2.0-py2.py3-none-any.whl (30 kB)  
 Installing collected packages: funcy, pyLDAvis  
 Successfully installed funcy-2.0 pyLDAvis-3.4.1

```

1 import pyLDAvis
2 import pyLDAvis.gensim
3
4 pyLDAvis.enable_notebook()

```

```
5 vis = pyLDAvis.gensim.prepare(lda_model, corpus, dictionary)
6 vis
```

 /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically and should\_run\_async(code)

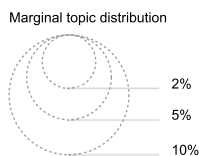
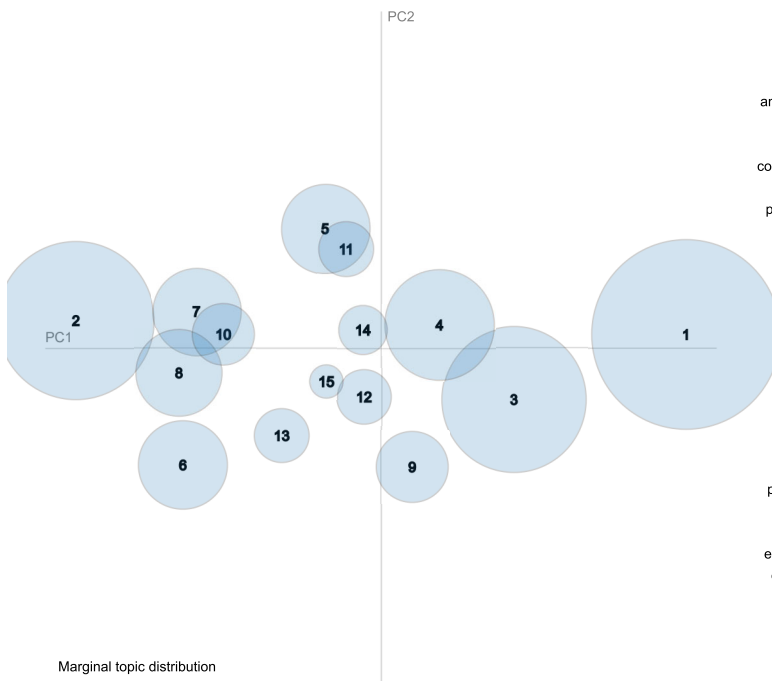
Selected Topic:

Slide to adjust relevance metric:<sup>(2)</sup>

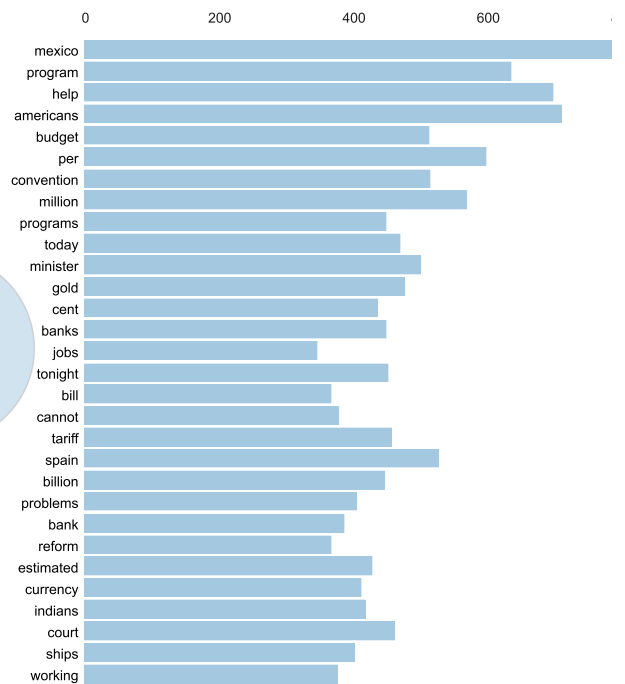
$\lambda = 1$

0.0 0.2 0.4 0.6

Intertopic Distance Map (via multidimensional scaling)



Top-30 Most Salient Terms<sup>1</sup>



Overall term frequency


Estimated term frequency within the selected topic

1. saliency(term w) = frequency(w) \* [sum\_t p(t | w) \* log(p(t | w)/p(t)) for topics t; see  
2. relevance(term w | topic t) =  $\lambda * p(w | t) + (1 - \lambda) * p(w | t)/p(w)$ ; see Sievert & Shirley


## Lab part

- ID-to-word mapping:** after calling the doc2bow method, all words are represented by their IDs. Consequently, when you use the print\_topics method, only these IDs are displayed, making the output challenging to interpret and less meaningful. Therefore, Task #1 is to incorporate an ID-to-word mapping to resolve this issue.

```
1 import itertools
```

 /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically and should\_run\_async(code)


```
1 # get a word dictionary of encoded inputs -> {word: id}
2 token = dictionary.token2id
3
4 # swap keys and values of word_dict -> {id: word}
5 word_dict = dict((v,k) for k,v in token.items())
6
7 # example of id-token mapping from dictionary
8 dict(itertools.islice(word_dict.items(), 10))
```

 /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically and should\_run\_async(code)


```
{0: 'according',
1: 'admitted',
2: 'advancement',
3: 'affording',
4: 'agree',
5: 'allowed',
6: 'arrangements',
```

```
7: 'ascertained',
8: 'attended',
9: 'blessed'}
```

```
1 word_mapping = []
2 for sentence in corpus[:2]: # get list
3     word_freq = {}
4     for word_tup in sentence:
5         word = word_dict[word_tup[0]]
6         freq = word_tup[1]
7         word_freq[word] = freq
8
9     word_mapping.append(word_freq)
```

 /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically and should\_run\_async(code)

```
1 # example of id-to-word mapping output
2 # format in dictionary of {word: frequency} in each sentences
3 word_mapping[0].items()
```

 /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should\_run\_async` will not call `transform\_cell` automatically and should\_run\_async(code)

```
dict_items([('according', 1), ('admitted', 1), ('advancement', 1), ('affording', 1), ('agree', 1), ('allowed', 1), ('arrangements', 1),
('ascertained', 1), ('attended', 1), ('blessed', 1), ('blessings', 2), ('communication', 1), ('compensation', 1), ('competent', 1),
('concerned', 1), ('concur', 1), ('considerations', 1), ('constituents', 1), ('contained', 1), ('currency', 1), ('declaration', 1),
('deemed', 1), ('deeply', 1), ('defined', 1), ('degree', 1), ('deliberate', 1), ('deliberations', 1), ('depredations', 1), ('deserve', 1),
('designated', 1), ('desirable', 1), ('difficulty', 1), ('disregard', 1), ('distant', 1), ('duly', 1), ('effectual', 2), ('efficient', 1),
('encouragement', 2), ('engage', 1), ('enlightened', 1), ('entered', 1), ('entirely', 1), ('entitled', 1), ('estimates', 1), ('exertions',
1), ('expectations', 1), ('expediency', 1), ('expedient', 1), ('foreigners', 1), ('frontiers', 1), ('fulfill', 1), ('fund', 1), ('genius',
1), ('george', 1), ('happiness', 1), ('hostile', 1), ('incident', 1), ('indians', 1), ('indispensable', 1), ('inevitable', 1),
('inhabitants', 1), ('institution', 1), ('intercourse', 2), ('interesting', 1), ('introduction', 1), ('intrusted', 1), ('lawful', 1),
('lay', 1), ('learning', 1), ('legislature', 2), ('manufactures', 1), ('merit', 1), ('objects', 1), ('official', 1), ('oppression', 1),
('ours', 1), ('papers', 2), ('patriotism', 1), ('peculiar', 1), ('perfect', 1), ('pleasure', 1), ('post', 2), ('presents', 1),
('preserving', 1), ('proceeding', 1), ('producing', 1), ('promoted', 1), ('promotion', 1), ('prospects', 1), ('providence', 1), ('punish',
1), ('realize', 1), ('recommendation', 1), ('reflection', 1), ('reliance', 1), ('relieved', 1), ('render', 3), ('requisite', 1),
('resolution', 1), ('respecting', 1), ('resulting', 1), ('rising', 1), ('roads', 1), ('rule', 1), ('sanction', 1), ('science', 1),
('sentiment', 1), ('skill', 1), ('society', 1), ('soldiers', 1), ('southern', 1), ('speedily', 1), ('speedy', 1), ('supplies', 1),
('task', 1), ('tend', 1), ('tribes', 1), ('truly', 1), ('uniform', 2), ('useful', 1), ('valuable', 1), ('vigilance', 1), ('ways', 1),
('western', 1)])
```

- 2. Topic Analysis:** The initial visualization from pyLDAvis shows significant overlap among many topics. Additionally, when hovering over each topic, you may notice that several words appear across multiple topics, many of which may not even contribute meaningfully to the analysis. Thus, for Task #2, try tuning the LDA parameters or incorporate additional text preprocessing. Finally, present your analysis of the topics derived from this dataset.


Ans: I have try 10-13 clusters. I found that 10 cluster got the least overlapping on each clusters so, I will stick with this result. Insight I got from the clusters: every cluster got the topic about Mexico but, got some difference. The left side talk about relationship of Mexico with Americans. In other hand, right-side of y-axis mention more about Spain.

```
1 lda_model = models.LdaModel(corpus=corpus, num_topics=10)
2
3 vis = pyLDAvis.gensim.prepare(lda_model, corpus, dictionary)
4 vis
```

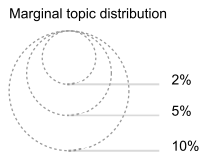
```
WARNING:gensim.models.ldamodel:no word id mapping provided; initializing from corpus, assuming identity
WARNING:gensim.models.ldamodel:too few updates, training might not converge; consider increasing the number of passes or iterations to impr
```

Slide to adjust relevance metric:<sup>(2)</sup>

$\lambda = 1$



0.0 0.2 0.4 0.6



Overall term frequency

Estimated term frequency within the selected topic

1.  $\text{saliency}(\text{term } w) = \text{frequency}(w) * [\sum_t p(t | w) * \log(p(t | w)/p(t))]$  for topics  $t$ ; see
2.  $\text{relevance}(\text{term } w | \text{topic } t) = \lambda * p(w | t) + (1 - \lambda) * p(w | t)/p(w)$ ; see Sievert & Shirley