Quantitative Text Analysis

Bruno Castanho Silva

Day 5 - Topic Models and What Else is out There?

Cologne Center for
Comparative Politics

Any questions from yesterday?

Topic Models

Topic models

- *“Probabilistic models for uncovering the underlying semantic structure of a document”* (Biel 2003)
- Goal: Identify similar documents that talk about similar things;
- Intuition: a text is composed of different topics; certain words are more associated with one topic than another, and can be used to identify the topics in a text, and how much of each is there.
- It is a **measurement strategy**

Examples

- Who talks more about immigration, Liberals or Conservatives?
- What do people talk about on Twitter at a given time?
- Social media posts with what kind of content are more likely to be banned in China?

The structure

- Documents are composed of different topics, in different proportions:
    - A speech by the education minister might be 70% education; 20% taxation; 8% family; and 2% immigration
- A **topic** is a **distribution** over a **fixed vocabulary**:
    - The *education* topic will have high probability for the words “university” and “teacher” and low probability for the word “Air-defense systems”
- Assumption: **topics** exist first, and **documents** are produced from those topics using the **words**

Words and documents

- Every word has a given probability of pertaining to **every** topic;
  - Highly discriminating words will have high prob for **one** topic and very low to all others.
- **All** documents contain **all** topics
  - But some might be (almost) entirely absent. In principle, all speeches can contain at least 0.001% education, but it may be that for some it is a round 0.

General logic

- Model to **generate** topics
- We must say in advance **how many** topics (*K*) there are in the corpus
- From that, models will work to assign probabilities for each word for each topic, and topic proportions

Breaking the dfm

- The dfm is broken down into two matrices:
- A document-topic matrix and a topic-word matrix

	K1	K2	K3	K
D1	1	0	0	1
D2	1	1	0	0
D3	1	0	0	1
Dn	1	0	1	0

	W1	W2	W3	Wm
K1	0	1	1	1
K2	1	1	1	0
K3	1	0	0	1
K	1	1	0	0

The two matrices

- Start with random assignment of words W to topics K, and of topics K to documents D;
- Rebuild the dfm (a *predicted* dfm). Will be terrible;
- Change topic assignment of word W_1; recalculate the document-topic matrix; rebuild the predicted dfm. Is it better?
- Change the topic assignment of word W_2; do the same as above. Keep going until convergence

Distributions

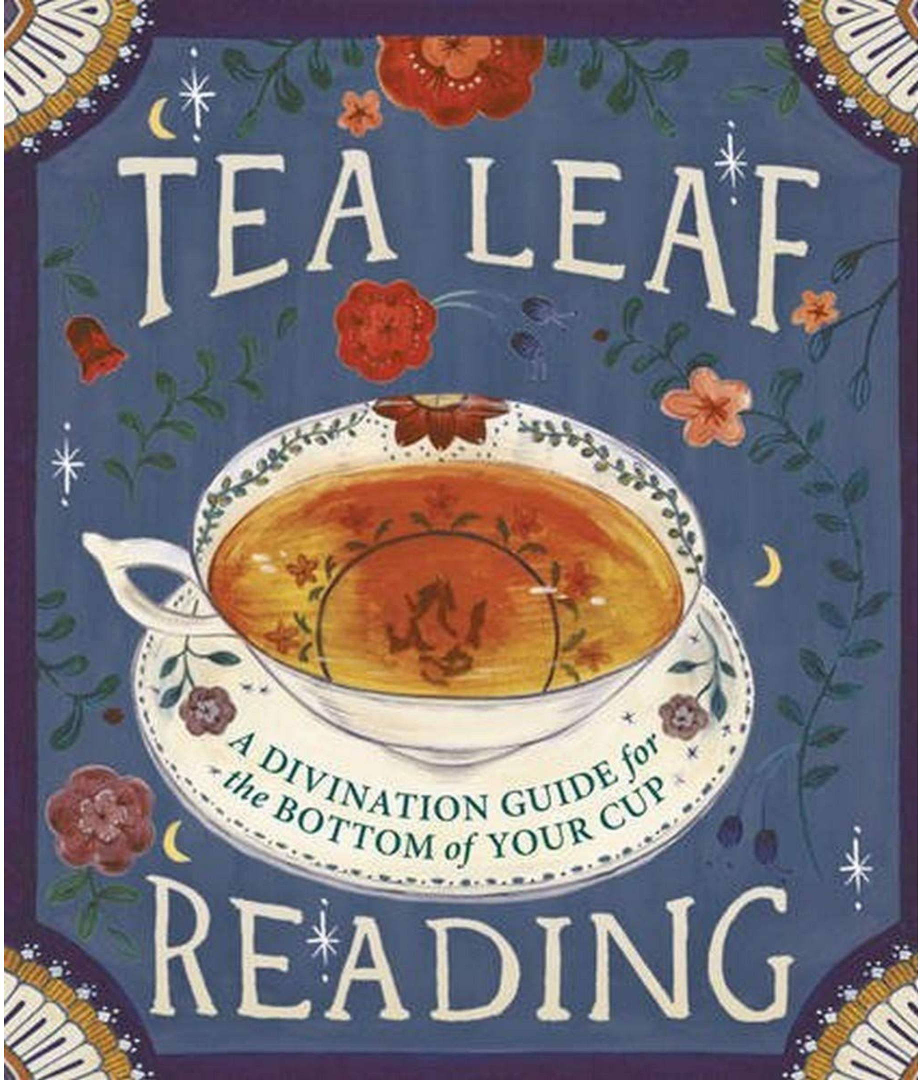
- We have to set the starting distributions:
  - Number of *k*. This is fixed
  - Topic-distribution across documents. Can be uniform (i.e., = 1, same proportion of topics in each document), or other forms: lower than 1 = each document is one topic; or symmetric (some documents talk about one topic, some talk about multiple);
  - Word-topic distribution: uniform distribution of words on topics or some topics have more words?
  - The latter two are optimized by the algorithm itself. But setting a reasonable one from the beginning accelerates estimation

What we get out

- For each topic *K*, a list of words most associated with it;
- For each document, the frequency of each topic;

What we **don’t** get out of it

- Interpretation. It will show you a list of words. It’s **your task** to figure out what they mean or what topic it is
- The ordering of topics is **random**. Topic 1 is **not** e.g. the first principal component.
- A clean way to select K.


TEA LEAF
A DIVINATION GUIDE for
the BOTTOM of YOUR CUP
READING

Criticism

- It’s not very interpretable. Oftentimes, several topics will not seem to have any substantive meaning;
    - You assign a “topic” based on a bunch of words, which other people might interpret differently.
    - E.g. what’s the topic: *hospice; benefits; payments; commuters; parenting; adjustment; project; approach; reports?*
- Makes it **very** a-systematic and hard to reproduce
- Resource-intensive. Models may take a long time to run - makes it problematic when we have to estimate for several values of K

Adding covariates

- We know our texts are not completely independent from one another;
  - Speeches given by members of different parties; during different times/legislatures; stories by different newspapers; etc
- How does that affect the frequency of words and topics in documents?
- Enter **Structural Topic Models**

Structural Topic Models

- **Content covariate**: Document-level covariate that predict variations in word frequency within topics
    - Each party uses different words to talk about the same topic (e.g., health care)
- **Frequency covariate**: Document-level covariate that predicts variation in frequency of topics across different documents.
    - Some parties talk more frequently about some issues (immigration).

It’s a regression

- At the end of the day, it’s a regression: how does the covariate you specify (e.g. party) affect the frequency of topics (e.g. immigration, economy, health, miscellaneous) in speeches?
  - We even get a point estimate and standard errors
- Or how does being in the treatment or control group affect answers to an open-ended survey question?

Let’s see this in R

Embeddings

So far we ignored…

- Similarity between words;
    - We consider “hospital” and “clinic” to be as similar to each other as they are to “croutons”
- Context
    - If we see the word “party” we don’t make a difference whether the original sentence was:
        - The Labour **Party** platform of 2015 was very ambitious
        - The **party** at Downing Street during lockdown is now troubling the PM

Word embeddings

- Word embeddings assign a vector of **values** for each word in relation to every other word in a corpus.
- Imagine we have the following words:
  - parliament; congress; committee; taxes; regression.
- In the methods we saw so far, the numeric representation of "parliament" in relation to the others would be
  - 1; 0; 0; 0; 0
- Because we're only looking at exact matches

Word embeddings

- Embeddings attribute a range of values to the other words, so that higher values indicate closeness to the term.
- So for the list
    - parliament; congress; committee; taxes; regression
- It might be
    - 1; 0.9; 0.6; 0.2; 0.05
- Suddenly, for each word, we have a vector representing the closeness of every other word in the vocabulary

Two main usages

- Data preparation for further analysis
    - Such as machine learning or scaling
- Semantic analysis
    - What words are most associated with certain terms of interest?
    - For example, look at all newspaper articles from 2010 to April 2022. What words were most associated with “refugee” from 2010-15, and then from 2015-Feb 2022? What does that tell us about media framing?

How do we get them?

- Thousands of different models to obtain embeddings
- Earlier/simpler versions: using only word counts and frequencies, and dimensionality reduction (e.g. PCA).
  - See a nice example application in R here: https://juliasilge.com/blog/tidy-word-vectors/
- Most recent, predictive models. Two popular approaches
  - Word2Vec
  - GloVe

How do we get them?

- Naturally, context of the corpus is very important
    - Word associations will be different if you train a model from parliamentary speeches or from sports blog posts
- Two options:
    - Pre-trained models. Several available, often trained in large corpora of e.g. Wikipedia articles
    - Train yourself, if you want context specificity (e.g., parliamentary speeches, articles on a given topic, etc)

What is needed

- If training your own: a very large corpus
  - E.g., Rheault and Cochrane (2020), entire parliamentary corpus of US, UK, and Canada over a century
- Computational power
- Python

One step further: BERT

- State of the Art model for NLP, developed by Google
- Both word and sentence representations
- Multiple vectors per word, allowing context.
  - Word2Vec doesn't make a difference still between **party**gate and Labour **party**. Will be a mix of the two, because there's only one vector per word
  - BERT can have more than one vector, thus taking in different meanings
- Also allows for out-of-vocabulary representations
  - Word2Vec doesn't know how to treat words it hasn't seen in the training data. BERT assigns a value based on the context.

In summary: What did we learn?

- Political text is an important source of information
- We can analyse texts descriptively. Which candidates are more similar? What words are more common?
- We can classify texts based on dictionaries
- We can use scaling methods to put our texts/speakers into dimensions
- Machine learning can help us classify texts into categories without defining dictionaries beforehand
- Topic models help us figure out what text are talking about