Computational Social Science

Topic Modeling II

Dr. Thomas Davidson

Rutgers University

October 30, 2025

Plan

1. Course updates
2. Structural Topic Modeling (STM)

Course updates

Homework 3

- Homework 3 on NLP will be released today on Canvas/Github
  - Due next Thursday at 5pm
  - Covers intro to NLP, word embeddings, and topic modeling

Topic modeling

- Topic modeling is an approach for understanding themes in documents
- Topics capture words that are frequently used together
  - A topic is a distribution across a vocabulary
- A document contains a mixture of different topics
  - A document is a distribution across topics

Structural Topic Modeling

Background

- LDA assumes *topic prevalence* (frequency topic is mentioned) and *topic content* (the words used to discuss a topic) are constant across documents
- STM extends LDA by "allowing for the inclusion of covariates of interest into the prior distributions for document-topic proportions and topic-word distributions" (Roberts et al. 2014).
  - This allows analysis of how topics vary according to other factors, for example the treatment in a survey experiment may alter open responses.

Structural Topic Modeling

Topic prevalence

- Prevalence refers to the frequency distribution of a topic across documents
- As social scientists, we often want to see how a topic varies by some categorical variable of interest
  - Author (person, organization, publisher, political party, etc.)
  - Time (day, month, year, decade, etc.)
  - Demographics (age group, gender, race/ethnicity, etc.)

Structural Topic Modeling

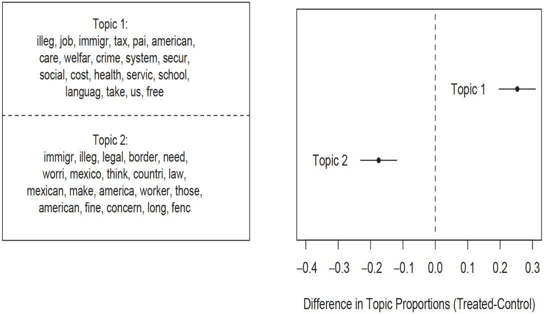
Topic content

- Content refers to the way different topics are discussed
  - As social scientists, we might expect different groups to use different kinds of language to refer to the same topic

Structural Topic Modeling

Analyzing open-ended survey responses using an STM

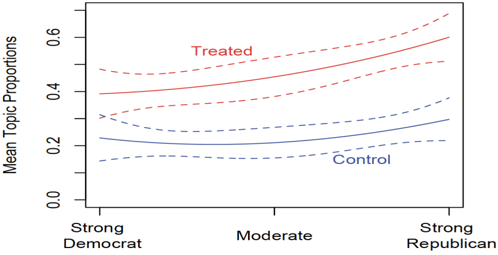
FIGURE 7 Words and Treatment Effect Associated with Topic 1



Structural Topic Modeling

Analyzing open-ended survey responses using an STM

FIGURE 8 Party Identification, Treatment, and the Predicted Proportion in Topic 1



Structural Topic Modeling

Loading the corpus

Loading a corpus of State of the Union speeches from 1900-2020. Each row represents a paragraph from a speech. 50% of paragraphs are sampled at random, otherwise models take too long to run!

```
library(tidyverse)
data <- as_tibble(read_csv("../data/sotu_texts.csv")) %>%
    sample_frac(0.5)
```

Structural Topic Modeling

Inspecting the texts

```
head(data$paragraph, n=3)
```

Structural Topic Modeling

Selecting metadata

```r
meta <- data %>% select(year, party)
head(meta)
```

Structural Topic Modeling

Preprocessing

The `stm` library has its own set of functions for processing data. `textProcessor` takes a corpus, plus metadata, and conducts pre-processing tasks. `prepDocuments` then converts the documents into the appropriate format.

```
library(stm)
#install.packages("stm")
processed.docs <- textProcessor(data$paragraph,
                                metadata = meta)

output <- prepDocuments(processed.docs$documents,
                        processed.docs$vocab,
                        processed.docs$meta,
                        lower.thresh = 10)
```

Structural Topic Modeling

Finding K

The STM package can calculate some heuristics for finding the "best" value of `K`. This can take a while as it must run each of the models specified in the vector passed to the `K` parameter.

```
library(parallel)
search.results <- searchK(output$documents, output$vocab,
                          K = c(20,40,60,80,100,120),
                          data = output$meta,
                          proportion=0.2, # proportion of docs held-out
                          cores=detectCores() # use maximum number of availabl
)
```

See https://juliasilge.com/blog/evaluating-stm/ for an alternative approach that enables some more post-estimation evaluation.

Structural Topic Modeling

Selecting K

```r
plot(search.results)
```

See Mimno, David, Hanna M Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. 2011. "Optimizing Semantic Coherence in Topic Models." In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, 262–72. ACL for discussion of the semantic coherence measure.

Structural Topic Modeling

Fitting a model

Fitting a model with `k=60`. The `party` variable is used as a covariate for both prevalence and content. Year is used as a covariate for prevalence, where `s()` is a non-linear spline function.

```
K=60
fit <- stm(documents = output$documents, vocab = output$vocab, K=K,
           data = output$meta,
           prevalence = ~ party + s(year), # s takes a non-linear funct
           content = ~ party, # content can only contain one variable
           verbose = TRUE
           )
```

Structural Topic Modeling

Storing/loading data

I stored the image of this workspace and uploaded it to Github. You can load the trained model and all other files in this script by running this line.

```
#save.image(file = "../data/sotu_stm.RData")
load(file = "../data/sotu_stm.RData")
```

Structural Topic Modeling

Inspecting the results

We can directly plot the proportions to show how frequent different topics are. Here are the first 20.

```
plot(fit, type = "summary", topics = 1:20)
```

Structural Topic Modeling

Inspecting the results

```
plot(fit, type = "summary", topics = 21:40)
```

Structural Topic Modeling

Inspecting the results

```
plot(fit, type = "summary", topics =41:60)
```

Structural Topic Modeling

Inspecting topics

```
labelTopics(fit, topics=14, n=10)
```

Structural Topic Modeling

Analyzing documents

We can use `findThoughts` to identify documents with a high weight in a given topic. Note that the original texts column does not work, I have to use the index for the metadata file to identify relevant columns.

```
t=14
thoughts <- findThoughts(fit, texts = as.character(data[as.numeric(rown
for (i in unlist(thoughts$docs)) {print(i)}
```

Analyzing documents

```
t=10
thoughts <- findThoughts(fit, texts = as.character(data[as.numeric(rown
for (i in unlist(thoughts$docs)) {print(i)}
```

Structural Topic Modeling

Analyzing documents

```
t=18
thoughts <- findThoughts(fit, texts = as.character(data[as.numeric(rown
for (i in unlist(thoughts$docs)) {print(i)}
```

Structural Topic Modeling

Estimating relationship between topic prevalence and metadata

```
prep <- estimateEffect(~ party + s(year), fit, meta = output$meta)
```

Structural Topic Modeling

Topic prevalence by party

Structural Topic Modeling

Prevalence over time

We can use the `year` variable to track how prevalence changes over time.

Structural Topic Modeling

Content by party

Structural Topic Modeling

Content by party

Structural Topic Modeling

Content by party

Structural Topic Modeling

- Resources
  - The STM website contains information on various tools and research papers that use the approach
    - There are several packages including `stmBrowser`, `stmCorrViz` and `stminsights` that enable more interactive visualization.
  - The vignette provides a closer description of the methodology and a hands-on guide to using the `stm` package.

Summary

- Topic modeling is an inductive approach for the summary of large text corpora
  - Analysis of topic models involves the interpretation of topics
  - A key challenge is selecting an appropriate number of topics
- LDA algorithm summarize as corpus into K topics
  - Each document is composed of a mixture of topics
  - Each topic is a mixture of words
- STM improves on LDA by allowing topic prevalence and content to vary by covariates
  - This is particularly useful for social scientific applications