Lab 13 - LDA and STM Models

Colin Case, with some code adapted from Rachel Porter

11/30/2022

As we discussed a few weeks ago, text is a rich data source for studying certain political phenomenon among both elites and mass public. In general, as political scientists, there are two general questions we can focus on, as discussed by (Rodriguez and Spirling (2022))[https://doi.org/10.1086/715162]:

1. Estimating political attention or the focus different populations of interest place on different aspects of policy or politics at different times.
2. Characterizing the nature of political conflict or the way in which different populations of interest use different language to discuss similar issues.

In answering the first type of question, topic models have proven extremely useful and will be our focus today.

Latent-Dirichlet Allocation Model

A few weeks ago, we started by working with how to download, collect, and clean different types of text. Now, we face the problem of actually being able to (1) measure certain characteristics in text and (2) draw inferences from text. To explore what latent topics are in the text, we can fit Latent-Dirichlet Allocation (LDA) models.

What the LDA model is doing, is finding the following parameters using the Bayesian approach:

$$\varphi_{k=1,\dots K} \sim Dirichlet_V(\beta)\theta_{d=1,\dots M} \sim Dirichlet_K(\alpha)z_{d=1,\dots M,w=1,\dots N_d} \sim Categorical_K(\theta_d)w_{d=1,\dots M,w=1,\dots N_d} \sim Categorical_V$$

That is a lot of variables and parameters. Some quantities of interest include:

K - the number of topics, must be manually specified. V - the number of words in the vocabulary (tokens). M - the number of documents in the corpus. $N_{d=1,\dots M}$ - number of words in document d. N - total number of words in all documents, or $N = \sum_{d=1}^{M} N_d$ α - parameter of the Dirichlet prior on the per-document topic distributions β - parameter of the Dirichlet prior on the per-topic word distributions θ_i - topic distribution for document i φ_k - word distribution for topic k

So for us, we are mostly interested in θ_i and φ_k, or in plain English, what topics can characterize a particular document i, and what words can characterize a particular topic k.

Text Pre-Processing

Before we get to fitting an LDA model, we first need to complete some text pre-processing steps. The data you will be working with today are campaign positions from candidates websites from 2018 primaries that broadly relate to gender issue positions. We also document level covariates for each candidate's statement. Let's take a look and clean this data:

```r
# Clear Environment
rm(list = ls())

# Load Packages
library(quanteda)
library(ggplot2)
library(stm)
library(keyATM)

# Set working Directory
setwd("C:/Users/colin/poli787-fall22/lab13-text2")

# load in data on campaign positions
load("positions_data.RData")

# View Data
head(data)
```

As you can see, we have some statements without issue positions, as well as certain tags we need to remove like we saw a few weeks ago. First, remove these tags as well as any numbers or words not beginning with a letter. Then, remove any observations that don't have any text in the trimmed field. Finally, create a corpus object, remove stopwords, stem words, as well as remove punctuation.

```r
## remove any tags, unwanted text for analysis
data$trimmed <- gsub("[\n]", "", data$trimmed)
data$trimmed <- gsub("d5", "", data$trimmed)
data$trimmed <- gsub("d0", "", data$trimmed)
data$trimmed <- gsub(" *\\b[[:alpha:]]{1,2}\\b *", " ", data$trimmed)

## remove numbers, any words that do not begin with a letter
data$trimmed <- gsub('[0-9]+', '', data$trimmed)
data$trimmed <- gsub("[^a-zA-Z]", " ", data$trimmed)

## drop any observations with no text
data <- subset(data, !is.na(data$trimmed))

## create corpus object, set text field
issues <- corpus(data, text_field = "trimmed")

## finish preprocessing steps (stop words (including Congress), stem and remove punctuation)
eng.stopwords <- __________________________
eng.stopwords <- c("_______", _______)
iss_dfm <- dfm(issues,
               _______,
               _______,
               _______

## Examine top features
topfeatures(iss_dfm, 100)
```

Now that we have our cleaned text, we need to prep the documents for estimating the LDA and STM later in the lab. We also need to make a decision about what our threshold is for dropping certain observations that don't have a lot of text, as well as dropping infrequent words. For this, we'll go with a vocab threshold of 5, but keep in mind this really depends on your research question and what the data looks like (the same goes with the pre processing decisions above).

Please note, for this lab and for the sake of time, we are going to be estimating an LDA using the `stm` command without covarariates. It will be similar to an LDA estimated using a package designed specifically for LDA, but in general you should use a command and document preparation specific to LDA if that is your sole focus (for an explanation of the differences, see Roberts et al. 2014).

```
# Convert dfm to stm and lda structure
iss_lda <- convert(______, to = '____')
iss_stm <- convert(______, to = '____')


# Plot varying drop thresholds
plotRemoved(iss_stm$documents, lower.thresh=seq(1,200, by=20))

# Determining drop threshold
iss_lda <- prepDocuments(iss_lda$_______, iss_lda$______,
                         iss_lda$meta, lower.thresh = 5)

iss_stm <- prepDocuments(iss_stm$_______, iss_stm$_____,
                         iss_stm$meta, lower.thresh = 5)
```

Number of Topics

This is where our researcher determination becomes very important the choices we make here will have consequences on our results. I would suggest replicating your analysis across multiple topic/drop thresholds to ensure results are not model dependent. Per Grimmer and Stewart (2013), there is no surefire way to choose your number of topics; but there are data-driven approaches for making this determination. The `searchK` function allows for the repeated estimation of stms across different k-topic thresholds. These repeated estimations can then be compared across several diagnostic values to determine K.

To do this, we can use a variety of metrics from searchK (note, this will take a while; it can also be done with `FindTopicsNumber`):

1. Held-out likelihood — the probability of held-out documents given a trained model; goal close to 0 essentially — how good is the model at predicting held out words?
2. Residuals — If the model is correctly specified. If it correctly specified the multinomial likelihood the dispersion of residuals sigma2 should be 1; If > 1, too few topics, cannot account well for topic over-dispersion essentially — is there too much variability in topics?
3. Lower-bound — approximation to the lower bound on the marginal likelihood. You can think of it as the model's internal measure of fit. Goal is to MAXIMIZE
4. Semantic Coherence — Semantic coherence is maximized when the most probable words in a given topic frequently co-occur together. Goal is to MAXIMIZE.

```
# Use searchK to find potential number of topics, K, between range of 8 and 12
model_search <- searchK(iss_lda$_____,
                        iss_lda$____,
                        K = _______,
                        data = iss_stm$meta,
                        seed = 1996)
```

```
# Plot Results
plot(model_search)
```

As you can see, not all metrics consistently point to the same topic number. This is why looking at results across multiple different topic numbers is required. Looking across the different measures, 9 seems to perform the best when looking at held-out likelihood (closest to zero), residuals (closer to 0, but maybe suggests we need more topics), higher semantic coherence, and although worse when it comes to lower bound.

Fitting the LDA

Once you've found the best K to use, you can now fit the LDA. Because we are going to spend most of this lab on topic modeling, and this is a more common tool than just running an LDA, we'll use `stm` without covariates specified.

```r
# Set seed for consistent topic ordering
set.seed(1996)
# Estimate LDA model using STM without covariates
initial_lda <- stm(documents = ___________,
                   vocab = ___________,
                   K = __,
                   data = iss_lda$meta,
                   init.type = "Spectral")
```

What is init.type? The spectral approach utilizes the connection of LDA with non-negative matrix factorization that provides theoretical guarantees that the optimal parameters will be recovered. This overcomes issue of local modes, which may cause differences in results based on changes in starting values (i.e. different seeds set). Spectral initialization should produce more consistent results across multiple runs; setting the seed becomes unnecessary.

Now that we have the model, we can look at the topic prevalence as well as the words that most frequently occur in that topic. By running `labelTopics`, we can get the following:

1. Highest Prob: Words with the highest probability of ocurring in a given topic
2. FREX words: Highest probability words re-weighted by how exclusive they are to a given topic
3. Lift words: Words divided by their overall frequency in other topics
4. Score words: Divide log frequency of a word in a topic by log frequency in other topics

```r
# View Topics and top words
labelTopics(________, n = 10)
# Save Topic proportions
theta <- data.frame(initial_lda$_____)
```

Something you should note is that topics are not always clear. There is certainly a post-hoc labeling process that occurs when deciding what topics are. There is some great work on standardizing the process of coming up with topic labels (see Ying, Mongomery, and Stewart 2021 as well as below for a apriori approach using Keyword Assisted Topic Models). For example, topic 4 and topic 7 are somewhat similar. Same for topic 5 and 9.

From the looks of the top words, we can use the following labels: (1) Women's Rights (2) Family Leave (3) Abortion (4) Reproductive Healthcare (5) Economic Opportunity (6) Body Autonomy (7) General Healthcare (8) Violence against Women (9) Equal Pay (note: this are quickly done and probably a great example of how labeling can at times be done hastily).

Let's now look at the frequency and the topic quality. As you can see, topic 1 is the most discussed by candidates while topic 8 is the least. When looking at the topic quality, we can see which topics are of higher quality. Topic 1 in particular has relative high exclusivity (words most likely to occur in this topic are not occurring in other topics) and semantic coherence (most probable words appear together frequently). This matches the anecdotal observation of what we discussed above regarding topics 4 and 7 being somewhat similar as well as topics 5 and 9.

```r
# Plotting overall topic frequency
plot(initial_lda, type = "summary", xlim = c(0, .5))

# Plotting overall topic quality
topicQuality(initial_lda, documents=iss_stm$document)
```

```r
# Making that plot a little better
labels <- c("Women's Rights", "Family Leave", "Abortion", " Reproductive Healthcare", "Economic Opportu

semcoh <- semanticCoherence(_________, documents = iss_stm$documents)
exclusivity <- exclusivity(_________)

semco <- data.frame(______, ______, ________)

ggplot(semco, aes(x = semcoh, y = exclusivity, label=labels)) +
  geom_point() +
  geom_text(aes(label=labels),hjust=-.05, vjust=1) +
  ylim(8,10) +
  xlim(-70,-15) +
  theme_bw() +
  xlab("Semantic Coherence") + ylab("Topic-Word Exclusivity") +
  theme(text = element_text(size=18))
```

Exercise: Estimating STM

We are now going to go through the same process as above, however this time we will be working with an STM rather than just an LDA model. For this, we are going to test whether or not candidates of different gender focus on different topics when discussing women's issues.

We already set up the df to be used (`iss_stm`). This time, go through the steps of (1) selecting the number of topics (in the range of 8 to 12) while specifying the prevalence argument as ~ gender + as.factor(ss_party) + openseat (2) estimate the STM using the same argument as above with the prevalence specified (3) label and plot the topics (4) test hypothesis regarding differences in topic prevalence across candidates of different genders.

```r
# Find K using SearchK
model_search <- ________________
```

```r
# Plot SearchK results
plot(model_search)
```

```r
# Estimate STM including document covariates
initial_model <- ________________
```

```r
# View Topic top words and general prevalence
____________________

____________________

# Plotting overall topic frequency
_________________________________

# TESTING HYPOTHESIS (Gender)
# Making Topic Labels Based on Top Words
labels <- c("Women's Rights", "Family Leave", "Abortion", " Reproductive Healthcare", "Economic Opportu

prep <- estimateEffect(___ ~ ______, ___________, meta = iss_stm$meta)

plot(_____,
     covariate="_____",
     c(1:9),
     model=_________, method="difference",
     cov.value1="0", cov.value2="1", ci.level = .90,
     main = "Relationship between Gender & Issue Coverage",
     xlab = "Greater Female Candidate Coverage ... Greater Male Candidate Coverage",
     xlim=c(-.15,.15), labeltype = "custom", custom.labels = labels)
```

As we expected, there are significant differences in how candidates discuss women's issues!

Keyword Assisted Topic Models

As we discussed above, labeling topics after the fact is somewhat of an imprecise science. Further, you may approach certain research questions and already have an idea of certain topics that may be important, but those topics don't appear readily in the results of the STM or LDA. For example, say we want to focus on how candidates of different genders discuss cancer screenings specially when it comes to women's health. As we can see, topic 7 discusses mammograms but is covering more than just this topic.

This is where keyword assisted topic models come in. KeyATM allows the researcher to specify keywords beforehand that define specific topics. In the simplest terms, this places more weight on these words when defining the topic and has been shown to produce more semantically coherent topics (See Eshima, Imai and Sasaki 2019 for more details).

For this lab, we are going to focus on creating specific topics surrounding different aspects of women's healthcare: a general healthcare topic, planned parenthood, birth control, abortion, and cancer.

```
# Convert dfm to keyATM structure
keyATM_docs <- keyATM_read(texts = _______)
summary(keyATM_docs)

# recalling top words across documents
topfeatures(iss_dfm, n = 100)

# specify keywords for topic modeling
keywords <- list(Healthcare             = c("healthcar", "afford", "health"),
                 Parenthood             = c("plan", "parenthood", "access"),
                 Birth_Control          = c("contracept", "birth", "control", "pill"),
                 Abortion               = c("abort", "safe", "legal", "pregnanc", "matern"),
                 Cancer                 = c("breast", "cancer", "screen", "access"))

# visualize keyword occurrence across documents; determine strength of keywords
_____________(docs = __________, keywords = _________, label_size = 5)

# specify covariate data for modeling
covariate_data = data[,c("gender", "openseat", "ss_party")]
covariate_data$ss_party <- as.factor(covariate_data$ss_party)

# specifying topic model
out <- keyATM(docs              = ________,
              no_keyword_topics = ________,
              keywords          = ________,
              model             = "________",
              model_settings    = list(covariates_data = ________,
                                       covariates_formula = ~ ________________),
              options           = list(seed = 250, iterations = 1000))

# assessing topic content
```

```r
top_words(____, n=15)

# assessing model fit
plot_modelfit(____)
plot_pi(____)

# examining our hypothesis
strata_topic <- by_strata_DocTopic(out, by_var = "_______",
                                   labels = c("Male", "Female"))
fig_doctopic <- plot(strata_topic, var_name = "______", show_topic = c(1:5))
fig_doctopic
```