Data Analysis with Topic Models for Communications Researchers

Todo list

1 Abstract

We present a non-technical introduction to data analysis with topic models for communications researchers. We motivate the discussion with a research question from social media communications research. We then discuss statistical aspects of topic models as we illustrate these methods with data from The New York Times. We complement our discussion with computer code (in the R computing language) that implements our methods. We close with thoughts about the future value of topic modeling to communications researchers.

2 Motivation

We are in the big data era. Social media inundates us with status updates and tweets, while bloggers share their views on current events. These new media interact with more established media, such as print news media, television, and radio. What do they have in common? One answer is that they all can be viewed as data sources in which words - whether written or spoken, tweeted or blogged - are the data.

Can we write a scenario like that of Blei's

Think about how we currently browse media. Let's suppose that we want to read about mass shootings. We might search for the key words "mass" and "shooting". Doing so, we will find some articles that deal with mass shootings, but other results may deal with other types of "shootings", such as film shooting or accidental shooting. Even if we specify that the two-word phrase "mass shooting" must appear in our results, we still may find some articles that merely mention mass shootings without focusing on one or more mass shootings.

Blei (2012) envisions an alternative search strategy in which we search for articles that are *about* "mass shootings". Some of the resulting articles would have the phrase "mass shooting", but containing that two-word phrase is not required; the sole requirement is that the article focus on one or more mass shootings. That is, we'd like to see all of the articles that have the theme, or "topic", mass shooting. Topic modeling is one method that may ultimately enable us to do the theme-based browsing that Blei (2012) suggests.

"Topic modeling" refers to a collection of statistical and machine learning methods that have the goal of discovering latent topics in a large collection, or "corpus", of texts. The tremendous rise in computing speed and memory capacity, coupled with the increasing availability of digitized texts, has enabled researchers working at the interface of quantitative methods and social sciences to treat written texts as data. While data analysis of documents is still in its infancy, scientists nevertheless have made great progress towards computational dissection and interpretation of texts. We detail below, with limited use of statistical terminology, how these methods work and why they may be useful in communications research. We illustrate topic modeling methods in the analysis of all New York Times articles from three days in March 2016. We also provide an appendix with computer code (in the R programming language) that implements these methods.

3 Background

LDA is an extension of an earlier text analysis method called "probabilistic latent semantic analysis" (pLSA) (Hofmann, 1999). Hofmann (1999) described pLSA as a "latent class model for factor analysis of count data". The novelty of LDA, compared to pLSA, is that LDA places a Dirichlet process prior on the probability distributions of words. This use of a Dirichlet process prior makes computations convenient because of a Bayesian statistical concept called "conjugacy". That is, the posterior distribution, which is the object of statistical inference, is also a Dirichlet process.

In the 13 years since the publication of Blei, Ng, & Jordan (2003), researchers have developed a wide variety of extensions to LDA. Many of these extensions of LDA accommodate relationships among topics. Widely used extensions of LDA include correlated topic models (Blei & Lafferty, 2007), hierarchical topic models (D. Griffiths & Tenenbaum, 2004), author-topic models (Rosen-Zvi, Griffiths, Steyvers, & Smyth, 2004), and dynamic topic models (Blei & Lafferty, 2006). Looking towards the future, researchers will need to fine tune and extend existing methods to accommodate new text data structures. One current research frontier is in topic modeling of tweets on Twitter. Twitter data present challenges because they are streaming and each message is no more than 140 characters.

Topic models have found applications in a wide range of disciplines. Pritchard, Stephens, & Donnelly (2000) used a statistical model much like LDA with genetic marker data to infer population structure and relatedness in human subjects. DeRose, Roy, & Boehm (2014) and Roy, DeRose, & Boehm (2013) used LDA (and other methods) in efforts to characterize themes from a collection of Victorian novels. Other examples include a 2008 study by Hall, Jurafsky, & Manning (2008), in which they applied LDA to examine the history of ideas. Grimmer (2010) applied a hierarchical topic model to study agendas in Senate press releases.

4 Box's Loop & Data analysis

The perspective that guides our data analysis is based on ideas of the University of Wisconsin-Madison statistician George Box and coworkers. Blei (2014) uses the ideas of Box and colleagues in the specific case of iterative refinement of topic models. Box (1976) articulated a process of scientific inquiry and statistical collaboration in which scientists put forth a hypothesis about the natural world, and, with assistance and guidance from statisticians, design experiments to test the scientific hypothesis. After statistical analysis of the experimental data, the scientific hypothesis is refined, which leads to design and implementation of another experiment, and the iterative process between scientific hypothesis and experiment continues. Box also suggested (CITATION) that a statistician, working in collaboration with scientists, might use scientific questions as motivation to develop new statistical methods in experimental design and data analysis. In this sense, there is a second iterative loop by which a statistical researcher develops novel methods because of their immediate need to answer a scientific research question. Blei (2014) adapts these iterative processes to the case of latent variable models, such as LDA and related models. He argues that we view the use of a topic model for a specific data analysis task as an iterative procedure in which one proposes a simple topic model for the data analysis, fits the model, interprets the results, and then, if needed, refines the topic model with the goal of achieving data analysis results that are more consistent with the research goals.

5 Latent Dirichlet Allocation

Blei et al. (2003) introduced a (generative) statistical model called "latent dirichlet allocation" (LDA) in 2003. Although others had described similar statistical models (Pritchard et al., 2000), Blei et al. (2003) first applied the statistical model to text analysis. Researchers in a wide range of disciplines - including genetics, linguistics, psychology, political science, and

others - have enthusiastically adopted LDA and related methods.

5.1 What is the intuition behind the model?

As Blei (2012) writes, the key to understanding LDA is to recognize that a given document - be it a research article, a novel, or a blog post - exhibits multiple topics. Each topic, in turn, is, in a technical sense, a probability distribution over words. For example, a topic related to evolution may heavily weight the words "evolution", "evolutionary", "biology", "phylogenetic", and "species". In a given collection of documents, which we term a "corpus" of documents, we assume that relatively few topics - on the order of 10 to 50 for most analyses - are present.

5.2 In what sense is the model 'generative'?

We say that LDA is a generative model because we specify a joint probability distribution that allows generation of observed samples. In a more precise manner, we specify a probability distribution over topics (again, where topics are themselves probability distributions over words) that is shared by the corpus. Each document can be viewed as a draw from this probability distribution. The observed topic (probability distribution over words) tells us how the words are distributed.

5.3 What is the model?

LDA models have a hiearchical structure in which words make up documents, and a collection of documents is a corpus. The corpus is assumed to have (unobserved) topics, or themes. The purposes of LDA, then, are to discover the unobserved topics from the texts and to characterize documents by the topics that they contain.

We distinguish generative models from discriminative models. A generative model, such as LDA, is one in which we specify a joint probability distribution for all variables (including those that are unobserved) and, thus, enable the creation of observations. On the other hand, a discriminative model relies on a conditional probability distribution and doesn't permit generation of samples from the joint probability distribution.

5.4 What are its assumptions?

5.5 What are its limitations?

The original LDA model However, LDA's flexibility and adaptability have reduced the impact of its initial limitations. In the last decade, researchers have devised extensions of LDA, such as "dynamic LDA", that enable one to model topic evolution over time. Another extension of LDA, which is known as "correlated LDA", accounts for correlations among topics.

6 LDA with statistical terminology

We can also describe LDA with more formal statistical terminology. We let $\beta_1, \beta_2, ..., \beta_K$ be the K topics, where each β_k is a probability distribution over the vocabulary. Topic weights (or proportions) we denote by θ_d for the d^{th} document. We let $\theta_{d,k}$ be the topic proportion for topic k in document d. Topic assignments for the d^{th} document are z_d, where $z_{d,n}$ is the topic assignment for the n^{th} word in the d^{th} document. Observed words for the d^{th} document are w_d, where $w_{d,n}$ is the n^{th} word for the d^{th} document.

We then use the above notation to specify the joint probability distribution for all variables in the model:

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D}) = \prod_{i=1}^{K} p(\beta_i) \prod_{d=1}^{D} p(\theta_d) \left(\prod_{n=1}^{N} p(z_{d,n}|\theta_d) p(w_{d,n}|\beta_{1:K}, z_{d,n}) \right)$$

The vertical bars in the last expressions denote conditional probabilities. Note that the topic assignments $z_{d,n}$ depend on the document's topic proportions. Whatismore, the observed word $w_{d,n}$ has a probability distribution that depends on both the entire collection of topics, $\beta_{1:K}$, and the topic, $z_{d,n}$, assigned to that word.

7 Example of LDA with print media

An example may help to illustrate our approach. Let's suppose that we want to understand the themes, or topics, that appear in a collection of print media articles. One strategy is to read every article in the collection. We could then assign one or more themes to each article and, at the end of our analysis, we would have a collection of themes. An alternative approach is to use topic modeling to identify topics, which we treat as themes, in the collection of articles.

To demonstrate our topic modeling approach to discovering themes, or topics, in print media, we used the Lexis Nexis Academic database to acquire 746 New York Times (print) articles from March 19, 20, and 21 of 2016.

We read the raw downloaded text files into R, then partitioned the files into the 746 articles. To each article, we applied a series of text processing steps. We split each article into its distinct words and removed white space and punctuation before converting all text to lower-case. At this stage, we had a collection of words for each of the 746 distinct articles. After removing words that occurred fewer than three times in the collection of articles and those that appeared on our "stop list" of frequently occurring short words, including articles and prepositions, we created the vocabulary and the document-term matrix. We then used existing R code (from the "lda" R package) to perform collapsed Gibbs sampling to sample from the posterior probability distribution of the parameters. Note that this computational implementation requires us to specify the number of topics in the collection. We arbitrarily chose 20 topics for our initial topic models.

8 Inference in topic models

Existing strategies for fitting topic models can be divided into two classes: 1. variational Bayes methods and 2. sampling methods. Variational Bayes methods try to approximate the posterior probability distribution by maximizing a lower bound. In this sense, variational Bayes methods approach topic model fitting from a computer science optimization perspective. Alternatively, sampling methods, such as those based on Markov chain monte carlo (MCMC) approaches, draw samples from (an approximation to) the posterior probability distribution and estimate distributional parameters by their empirical sample statistics.

Computational implementations of both classes of strategies are freely available for the R statistical computing language. We include code to use these free implementations in the appendix.

9 Interpreting results of topic modeling

Statistical inference for topic models yields estimates for topics (i.e., probability distributions over words), assignments of words (within documents) to topics, and weights of topics in each document.

The user must impose meaning on the topics. For instance, a topic may put heavy weights on the words “genetics”, “gene”, “regulation”, “DNA”, “transcription”, and “RNA”. The data analyst needs to identify the similarities among these words - namely, that they describe concepts related to genetics and molecular biology. Fortunately, topic modeling procedures often generate topics that can be summarized in one or two words once the data analyst has inspected the most heavily weighted terms (for a given topic).

10 Visualizing topics

Visualizing topic modeling results is an active area of research. We present below strategies that involve static and interactive displays. Word clouds are a widely used method for presenting topic modeling results. Unfortunately, the standard approach requires a distinct word cloud for each topic. For models with more than ten topics, manually examining wordclouds becomes unwieldy. Word cloud construction is straightforward and implemented with freely available software, such as the R package "wordcloud". For a single topic, the default method for creating a word cloud involves the identification of the most heavily weighted words in the topic. The user may choose to assign font sizes that are proportional to the weights of each word. One newly developed method, which is implemented in the LDAvis R package, uses a singular value decomposition of the fitted document-topic matrix to calculate principal components. Each topic is then plotted in two dimensions, where the axes represent the first two principal components. The LDAvis R package enhances this second approach by making the figures interactive with D3 javascript. For our analysis of 746 New York Times articles, in which our initial topic models featured 20 topics, we created wordclouds for each topic in the resulting models.

11 Results from NY Times Articles Analysis

12 Discussion

We see that the

13 Future directions

14 Online resources

We've posted our code, results, and manuscript on github.com. To access these materials, please visit this url: http://github.com/fboehm/aejmc-manuscript-2016

David Mimno, a Cornell University scholar, curates an annotated bibliography of topic modeling research(Mimno, 2016). His bibliography is available at this url: http://mimno.infosci.cornell.edu/topics.html

15 Computational implementation of LDA with R

We present below instructions and code for using LDA (with a collapsed Gibbs sampler) in the R statistical computing language (R Core Team, 2015). **Ask Dhavan about including code? Maybe as an appendix?**

Now we show all the code chunks:

```
library(knitr)
opts_chunk$set(echo = FALSE, message = FALSE, cache = TRUE)  # hide source code in the
install.packages("devtools")
devtools::install_github("fboehm/wordtools")
tx1 <- scan(file = "~/bret-research/twitter/aejmc-manuscript-2016/data/The_New_York_Time
    what = "character", blank.lines.skip = TRUE, sep = "\n", encoding = "UTF-8",
    skipNul = TRUE)
tx2 <- scan(file = "~/bret-research/twitter/aejmc-manuscript-2016/data/The_New_York_Time
    what = "character", blank.lines.skip = TRUE, sep = "\n", encoding = "UTF-8",
    skipNul = TRUE)
```

```
tx <- c(tx1, tx2)
library(wordtools)  #install my R package 'wordtools'
tx_list <- split_tx(tx = tx, patt = "Copyright 20")
library(stringr)
tx_list2 <- sapply(FUN = function(x) x[-(1:which(str_detect(string = x, pattern = "LENGT
    X = tx_list)
myfun <- function(x, pattern = "Web Blog") {
    collapsed <- paste(x, collapse = " ")
    !stringr::str_detect(collapsed, pattern = pattern)
}



indswb <- sapply(FUN = myfun, X = tx_list2)
indsurl <- sapply(FUN = myfun, pattern = "URL", X = tx_list2)
library(magrittr)
good_art <- tx_list2[indswb] %>% sapply(FUN = function(x) {
    x[-(which(str_detect(x, "URL")):length(x))]
})
library(tm)
stopwords <- c(tm::stopwords("SMART"), "dr", "mr", "ms", "mrs")  # add titles to stopli
good2 <- sapply(FUN = function(x) paste(x, collapse = " "), X = good_art) %>%
    stringr::str_split(pattern = " ") %>% sapply(FUN = function(x) gsub("'",
    "", x)) %>% # remove apostrophes
sapply(FUN = function(x) gsub("[[:punct:]]", " ", x)) %>% # replace punctuation with sp
sapply(FUN = function(x) gsub("[[:cntrl:]]", " ", x)) %>% # replace control characters
sapply(FUN = function(x) gsub("^[[:space:]]+", "", x)) %>% # remove whitespace at begin
sapply(FUN = function(x) gsub("[[:space:]]+$", "", x)) %>% # remove whitesp
```

```
sapply(FUN = function(x) tolower(x)) %>% sapply(FUN = function(x) x[!(x == "")]) %>%
    # remove elements that are ''
sapply(FUN = function(x) x[!(x %in% stopwords)])
# remove stopwords from
# http://cpsievert.github.io/LDAvis/reviews/reviews.html compute the table
# of terms:
n_min <- 3
term_table <- table(unlist(good2)) %>% sort(decreasing = TRUE)
term_table <- term_table[term_table >= n_min]
vocab <- names(term_table)
get_terms <- function(x) {
    index <- match(x, vocab)
    index <- index[!is.na(index)]
    rbind(as.integer(index - 1), as.integer(rep(1, length(index))))
}
documents <- lapply(good2, get_terms)
# Compute some statistics related to the data set:
D <- length(documents)  # number of documents
W <- length(vocab)  # number of terms in the vocab
doc.length <- sapply(documents, function(x) sum(x[2, ]))  # number of tokens per docume
N <- sum(doc.length)  # total number of tokens in the data
term.frequency <- as.integer(term_table)  # frequencies of terms in the corpus
# MCMC and model tuning parameters:
K <- 20
G <- 5000
alpha <- 0.02
eta <- 0.02
```

```r
# Fit the model:
library(lda)
set.seed(2016 - 3 - 22)
fit1 <- lda.collapsed.gibbs.sampler(documents = documents, K = K, vocab = vocab,
    num.iterations = G, alpha = alpha, eta = eta, initial = NULL, burnin = 1000,
    compute.log.likelihood = TRUE)
set.seed(2016 - 3 - 21)
fit2 <- lda.collapsed.gibbs.sampler(documents = documents, K = K, vocab = vocab,
    num.iterations = G, alpha = alpha, eta = eta, initial = NULL, burnin = 1000,
    compute.log.likelihood = TRUE)
set.seed(2016 - 3 - 23)
fit3 <- lda.collapsed.gibbs.sampler(documents = documents, K = K, vocab = vocab,
    num.iterations = G, alpha = alpha, eta = eta, initial = NULL, burnin = 1000,
    compute.log.likelihood = TRUE)

library(wordcloud)
for (i in 1:K) {
    cloud.data <- sort(fit1$topics[i, ], decreasing = TRUE)[1:50]
    wordcloud(names(cloud.data), freq = cloud.data, scale = c(3, 0.1), min.freq = 1,
        rot.per = 0, random.order = FALSE, col = 1 + i%%4)
}
for (i in 1:K) {
    cloud.data <- sort(fit2$topics[i, ], decreasing = TRUE)[1:50]
    wordcloud(names(cloud.data), freq = cloud.data, scale = c(3, 0.1), min.freq = 1,
        rot.per = 0, random.order = FALSE, col = 1 + i%%4)
}
```

```
library(wordcloud)
for (i in 1:K) {
    cloud.data <- sort(fit1$topics[i, ], decreasing = TRUE)[1:50]
    wordcloud(names(cloud.data), freq = cloud.data, scale = c(3, 0.1), min.freq = 1,
        rot.per = 0, random.order = FALSE, col = 1 + i%%4)
}
```

References

Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM*, *55*(4), 77–84.

Blei, D. M. (2014). Build, compute, critique, repeat: Data analysis with latent variable models. *Annual Review of Statistics and Its Application*, *1*, 203–232.

Blei, D. M., & Lafferty, J. D. (2006). Dynamic topic models. In *Proceedings of the 23rd international conference on machine learning* (pp. 113–120). ACM.

Blei, D. M., & Lafferty, J. D. (2007). A correlated topic model of science. *The Annals of Applied Statistics*, 17–35.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *The Journal of Machine Learning Research*, *3*, 993–1022.

Box, G. E. (1976). Science and statistics. *Journal of the American Statistical Association*, *71*(356), 791–799.

DeRose, C., Roy, C., & Boehm, F. (2014). Victorian eyes: Literary, statistical, and artistic perspectives on victorian novels – and dickens's unfinished murder mystery. *Significance*, *11*(2), 40–43.

Griffiths, D., & Tenenbaum, M. (2004). Hierarchical topic models and the nested chinese

restaurant process. *Advances in Neural Information Processing Systems*, *16*, 17.

Grimmer, J. (2010). A bayesian hierarchical topic model for political texts: Measuring expressed agendas in senate press releases. *Political Analysis*, *18*(1), 1–35.

Hall, D., Jurafsky, D., & Manning, C. D. (2008). Studying the history of ideas using topic models. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 363–371). Association for Computational Linguistics.

Hofmann, T. (1999). Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international aCM sIGIR conference on research and development in information retrieval* (pp. 50–57). ACM.

Mimno, D. (2016). Topic modeling bibliography. Retrieved from http://mimno.infosci.cornell.edu/topics.html

Pritchard, J. K., Stephens, M., & Donnelly, P. (2000). Inference of population structure using multilocus genotype data. *Genetics*, *155*(2), 945–959.

R Core Team. (2015). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from https://www.R-project.org/

Rosen-Zvi, M., Griffiths, T., Steyvers, M., & Smyth, P. (2004). The author-topic model for authors and documents. In *Proceedings of the 20th conference on uncertainty in artificial intelligence* (pp. 487–494). AUAI Press.

Roy, C., DeRose, C., & Boehm, F. (2013). Victorian eyes. Retrieved from http://victorianeyes.com

List of Figures

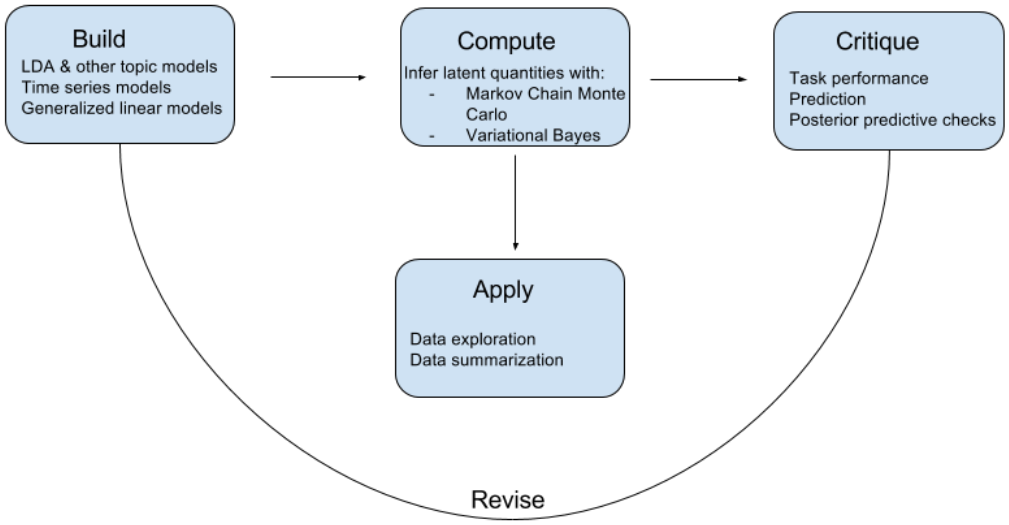

Figure 1: Iterative process for building, computing, critiquing, and applying topic models.

Figure 2: Wordcloud for one topic from a 20-topic model of 746 New York Times articles. Larger font size corresponds to greater weight of that word in this topic.

Figure 3: Wordcloud for one topic from a 20-topic model of 746 New York Times articles. Larger font size corresponds to greater weight of that word in this topic.

Figure 4: Wordcloud for one topic from a 20-topic model of 746 New York Times articles. Larger font size corresponds to greater weight of that word in this topic.

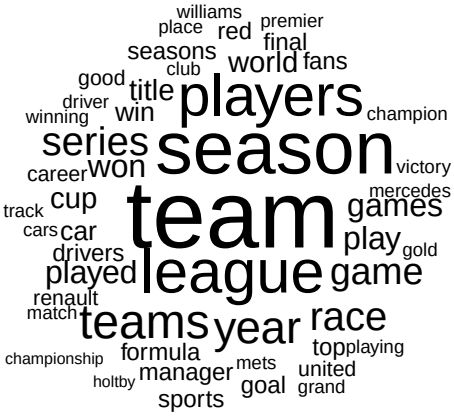

Figure 5: Wordcloud for one topic from a 20-topic model of 746 New York Times articles. Larger font size corresponds to greater weight of that word in this topic.