# Chapter 3: Identifying, Acquiring, and Representing Text Quantitatively [*]

Justin Grimmer[†] Margaret E. Roberts [‡] Brandon M. Stewart [§]

March 19, 2018

## 1 Introduction

Once you start to see text as a source of data, opportunities to study the social world are everywhere. For economists, insights about business management, economic growth, and marketing present themselves in company earning reports, consumer complaints, and product reviews. For political scientists, answers to representation, accountability, and political competition appear in speeches, press releases, and debates. For historians, valuable insights about the past are buried in archives and interviews. Even outside of social science, text data can facilitate learning in medicine and public health through medical records, manufacturing through accident reports, and the humanities through the digitization of literature.

How can we use this abundance of text data and burgeoning tools that facilitate the automated analysis of text to make inferences about the world? Collecting some text and running it through a statistical model is straightforward, but learning something about the human processes that the text represent is more complicated. As we discussed in Chapter 2, social science research is a sequential process and the appropriate way of collecting and

[*]Preliminary and incomplete draft, please do not circulate.

[†]Associate Professor, Department of Political Science, University of Chicago
[‡]Assistant Professor, Department of Political Science, University of California at San Diego
[§]Assistant Professor, Department of Sociology, Princeton University

analyzing texts will depend on where the researcher is within the scientific process and the specific problem the researcher wants to solve. In this chapter, we focus on how to set up the analysis of text with the intention of shedding light on a social science question. We begin by discussing how a principled way of gathering and sampling text is imperative for making inferences. After gathering text, we explain how text can be represented with numbers, and how the scientific question of interest is important in making decisions about how to represent text. We end with some examples of how different authors have made decisions to represent text differently in order to answer their own particular research question.

We stress that the process of gathering and pre-processing text data may need to be revisited several times during the research process. If we have never explored a corpus of text before and we do not know what it contains, our research question may change after looking at the data. The new research question may require a different representation of texts, or might suggest that we gather text from a wider or narrower range of sources. Because we are likely to revisit our initial choices at different stages of the research process, leaving out a set of the initial texts we collected as a validation set will help ensure that we can make systematic inferences at later stages of the research process. Therefore, we suggest that once a researcher has collected a corpus of text they leave some data aside or have a plan to collect more data for sequential testing.

## 2 Selecting Corpora

In this section, we provide a step-by-step approach to thinking through common sources of bias in text corpora. What corpus the researcher begins with will depend fundamentally on where they are in the research process. Once you've decided on a question, population, and quantity of interest, how well you can measure these quantities will depend on whether the texts within your corpus are representative of the population you are interested in. In this section, we provide a general check list of potential reasons why documents or texts might be missing from the corpus, from physical and incentive-based constraints on those providing and writing the documents to using statistical techniques or keywords to select subsets of a larger corpus.

## 2.1 What stage are you in the research process? What are the populations and quantities of interest?

Just as there is no best method for text analysis, as we described in Chapter 2, there is no "best" corpus of text. How useful a particular corpus is will depend on the question the researcher wants to answer. Many researchers begin their research process with only a vague idea of the question of interest. This is especially true in the analysis of text, where frequently analysts end up with a vast document set on their computer and very little idea of what these documents contain. Without digging into the data, the researcher may not know what they *can* measure, or how the corpus relates to social science theories in the area of their expertise. At this stage in the process, the analyst is in the discovery phase, she must explore, read, and describe the texts to better understand what questions they can answer and how they can be best used to make social science inferences about topics she is interested in. While the question of what the sample represents should always be in the back of her mind, she should continue revisit biases in the sample as the research question ossifies in her mind.

Other times, the researcher begins with a question of interest, which will make it more clear at the outset whether or not the texts are relevant. Questions of interest are typically in reference to a *population of interest*. For example, if my question of interest is "How do voters in the United States feel about the President?" my population of interest is voters in the United States. The question clarifies whether or not the corpus at hand reflects this population. For example, if we are using a Twitter corpus to answer "How do voters in the United States feel about the President?" we may become concerned that only certain types of U.S. voters use Twitter and therefore that this text sample does not reflect the whole U.S. voter population.

Questions of interest also suggest the ways that the texts should be compressed to be used in analysis, what we called a codebook function, or a $g$ function of how to represent text. We often refer to these as the *quantities of interest*. In our Twitter example, we might use the number of positive versus negative Tweets about the President as the quantity of interest, a task we will go through in more detail in the measurement chapter (Chapter 5). The population and quantities of interest guide how accurately the corpus can answer the question of interest.

Researchers in the social sciences ask questions that focus on wide variety of populations, domains, and time periods. For some research questions, text data may be used similarly to population surveys, where researchers hope to reflect the underlying opinions, activities, or demographics of an entire citizen population. For example, scholars have begun to use social media data like Twitter to describe everything from political opinions (O'Connor et al., 2010) to movie preferences (Asur and Huberman, 2010), to the spread of the flu (Lampos and Cristianini, 2010), and personal happiness in a population (Golder and Macy, 2011; Dodds et al., 2011). To accurately gauge underlying quantities of interest in the broader (both online and offline) population, these analyses depend on accurately accounting for the ways in which online populations are different from offline populations. Even within online populations, those who choose to engage in discussions about politics may be very different than those choosing to engage in discussions about entertainment, all which must be taken into account when generalizing to the population that does not discuss the topic within the data (Barberá and Rivero, 2014).

In other cases, the population of interest is a much smaller subset of the entire population. For example, Blaydes, Grimmer and McQueen (2018) explore the divergence in political thought between Middle Eastern and Christian societies. To reflect both Muslim and Christian political thinkers, they gather political advice texts from the 6th to the 17th centuries from both the Islamic tradition and Christian Europe. The authors' goal is not to gather *all* Muslim and Christian writings, or to reflect all thoughts on these subjects within these populations, but rather to reflect those with the most influence on political thought in this time period. In the digital humanities, analyses may be focused on exploring thematic trends in a particular genre, for example, Danish ghost stories (Abello, Broadwell and Tangherlini, 2012). Appropriate selection of texts in these cases will often rely on the ability to distinguish between stories that contain a subject like ghosts from those that do not.

Even if the corpus is not completely representative of the population of interest, there are times that we can still learn from it. Corpora that are known to be starkly biased and do not reflect the population of interest can still provide a window into a social processes. For example Gill and Spirling (2015) use historical records and leaked data to understand what types of information the United States government keeps secret and the process of declassification. Of course, the authors do not have access to *all* classified information, they only see that which has been leaked or that which has eventually been declassified. But in comparing information that is unclassified to that which was classified at some point, the authors can infer some of the types of topics that remain secret and can offer insight into what might be missing.

Once the researcher has a question, population, and quantities of interest, meaning that they have moved from the discovery to the measurement, causal inference, or prediction stage of the analysis, they should think carefully about the potential sampling biases in their corpus. In the subsequent sections, we provide a starting checklist of common sources of sampling bias that researchers might consider when they have a question and candidate corpus of interest. Not all of these sources of bias will pertain to all research questions and certainly some of the things we consider may be beneficial for answering some types of questions. However, we have found that for many questions and populations of interest, the four types of sampling bias below reappear frequently when analysts use text for social science inference.

## 2.2 Resource Bias

Texts are expensive to produce, gather, and collate. The first type of bias that we explore – *resource bias* – reflects the fact that because the analyses we discuss within this book rely on textual data, it will often better reflect populations with more resources to produce, record, and store texts. Every individual, institution, or group in the population of interest may not have the resources to record, preserve, and disseminate texts. Researchers should consider which portions of their population of interest have the capabilities to write and disseminate texts so that researchers can access them. These issues are especially salient when the population of interest expands beyond the texts themselves; for example, when researchers use text data produced by individuals to make inferences about a broader population of interest. Portions of the population may be illiterate, and these individuals will be completely unrepresented in texts. Texts downloaded from the Internet may underrepresent those who do not have access to the Internet; for example, some of the poorest areas in even the most developed countries are much less connected to the Internet and will be underrepresented in studies of social media (Norris, 2001).

Resources may also affect the availability of historical data. Historical events may be more likely to be recorded if they occur in areas where the press is present (Snyder and Kelly, 1977; Danzger, 1975). Archives might only contain documents that citizens found convenient to store and may be woefully unrepresentative of the whole set. Texts may have been lost, burned, or not stored, and stories may have never been transcribed. Communities and individuals with the capabilities to store and preserve texts will likely be very different than those without, and as such historical documents should not be considered a random sample of a larger set.

Government document availability may also reflect local resources. Some governments may have the personnel to transcribe meetings and make them available to the public, others will not. Even governments in the U.S. exhibit high variability in their capabilities to store and make available documents. For example, local police reports are sometimes handwritten, making them difficult to make available to researchers, while others are entered directly into a computer. The ways in which government documents are stored can make them more costly for researchers requesting them through Freedom of Information Act laws, as fees are charged to researchers for identifying and duplicating the documents of interest.

## 2.3 Incentive Bias

In some cases, not only resources but human incentives will affect the availability of documents and may explain corpus missingness. The ways in which humans' strategic writing and storing texts affects sampling bias we will call *incentive bias*. Individuals and organizations have incentives to fail to record, hide or destroy evidence that could cast them in a negative light. Individuals may be more likely to post social media that reflects the happiest or most successful aspects of their lives. For researchers studying interactions on social media, this selection bias may not be an issue, but for those who are trying to measure true emotional states of users, such selection may bias results. Similarly, politicians and governments may force removal of news and social media that undermines them, censoring others or self-censoring to frame political conversations (House, 2015; Boydston, 1991; Gup, 2008; King, Pan and Roberts, 2013; MacKinnon, 2008).

While transparency laws and initiatives may make texts more accessible to researchers, it also changes the incentives of those whom it affects. E-mail transparency within organizations may create incentives for members of these organizations to talk on the telephone about sensitive issues or hold informal meeting where discussions are not recorded. Transparency initiatives might make government leaders curtail their political participation if they are afraid of making missteps (Malesky, Schuler and Tran, 2012). When researchers are interested in using documents as a reflection of government internal workings, researchers should conduct extensive interviews with those writing, storing, and making documents available to understand the incentives behind document collection and the process through which the data are made available.

## 2.4 Medium Bias

The technologies and types of mediums in which texts are recorded will play an important role in the types of content that will be reflected in them; we will call this *medium bias*. Twitter, for example, previously only allowed 140 characters in one tweet, necessitating users to use abbreviation and links for further treatment of the topic. However, 140 characters has very different implications across languages — in denser languages like Chinese, users will be able to express more content in one post than in languages that necessitate many characters for each word like English. Further, the mode of expression in social media change with advances in technologies. As users have been able to use pictures, videos, and live steams online, the requirements of the accompanying text changes. Users interested in text analysis in social media should be aware of other types of metadata — links, pictures, and videos — when describing their content and how technological capabilities have changed over time.

The technology and evolution of the medium can create different cultures for users so that the text can only be understood in the context of the platform. Snapchat, a social media platform where posts are deleted immediately after they are read, will create different incentives for users than Twitter, where posts are public and more permanent. As most online platforms present posts to users that depend on their past history or their predicted characteristics, even within the same medium, users might have very different perspectives about what is being discussed on the platform because they are exposed to content based on a personalized algorithm. Politicians may try to steer the conversation toward topics that reflect better on them (Franco, Grimmer and Lee, 2016). Researchers should consider biases that might be produced from individual users' different experiences within the same platform and consider how the topics discussed might be strategically manipulated.

Text outside of social media is similarly influenced by its medium. Meeting transcripts or notes may reflect the content of the conversation, but not tone of voice, body language, or other forms of expressed emotion. Handwritten notes may contain drawings or doodles, computer written messages may contain emojis. When possible, researchers should read and interact with texts in their original context to understand how the social events of interest are translated by the medium into text, even if text analysis tools only take into account the text themselves.

## 2.5 Algorithmic Bias

As texts are sometimes selected using statistical methods, *algorithmic bias* can sometimes affect the selection of corpus. Say, for example, a researcher is interested in doing a quantitative analysis of ghost stories (Abello, Broadwell and Tangherlini, 2012). To do so, they might select every book in the library that includes in the title "ghost." However, this selection might be a biased sample of ghost stories. Some book titles that contain the word ghost may have nothing to do with ghost stories; for example, *Hungry Ghosts: Mao's Secret Famine* deals with the famine in China rather than with ghost stories themselves. Similarly, many books that don't contain the word "ghost" may still be ghost stories, *A Christmas Carol* is a ghost story without "ghost" in the title.

While these errors may at first seem random, they are often systematic. "Ghost" may orient the analysis toward one type of genre, missing books about phantoms which may have different themes. The word ghost might miss books about ghosts in other languages.

This type of keyword selection of corpuses is a frequent problem in the analysis of text because researchers often want to work with a focused corpus – like ghost stories – that is a subset of a larger, unwieldy corpus – such as all books. The problem is made worse by the fact that humans often have difficulty thinking of words off of the top of their head. They have limited recall, so may miss some of the most important words on which to select. Researchers might want to use methods that use the text itself to generate candidate keywords, or supervised learning methods that can identify texts of interest (King, Lam and Roberts, 2014; D'Orazio et al., 2014). Regardless, researchers should be transparent about how they selected the texts and what potential biases this might produce in what population the text reflects.

While keywords are a simple algorithm for selecting texts, more complicated algorithms are also used for selecting corpuses. Further, algorithmic bias is broader than a researcher's own method of selection as researchers often rely on third parties to provide data for them. Researchers who access data through Application Programming Interfaces (API) or other search functions should be as aware as possible of the underlying process that retrieves search results from queries. For example, some interfaces may account ignore capitalization, others might be sensitive to it. Some may provide a random selection of documents, others may return the most popular. The process through which the API makes documents available will affect the population the text represent and the research questions the text can answer.

## 2.6 Time-series problems with text selection

The biases in the selection of texts may change over time, making it difficult for researchers to discern whether shifts in the texts over time are created by changing opinions or styles of individuals within the population or by shifts in the sample. For example, when Facebook first started, it was originally open only to particular universities, but then was expanded to the public. Without considering the sample, a researcher might come to the conclusion that people have become less interested in education over the years. However, this change would be due to the changing sample of users rather than the actual shift in interest of the underlying population. Policy shifts in data transparency can create important discontinuities in government document samples that may appear to the researcher as if the substance of governance is shifting even if the underlying documents are static over time. Researchers should be careful not only to consider static bias but also how biases in document samples could vary over time to account for demographic, resource, medium, or incentive drift.

## 3 Representing Texts

After you have identified a corpus of interest that you think best represents your population of interest, the next step in analyzing the text is to retain only the necessary information from each document for your analysis. In other words, what information from the text do you want to measure, what will your quantities of interest be? Texts are complex – they encode meaning through word choice, phrasing, and the organization of concepts. There are many different features of text that an analyst might be interested in to encode as data. However, not all of the complexity of text is necessary for any particular analysis. Because quantitative analyses of text are often focused on characterizing a few general characteristics of a large document set – characterizing the haystack instead of a straw of hay – our analyses and quantities of interest will not typically require us to retain all information from the texts. Reducing the complexity of the text will be a key step in allowing the analysis to focus on only the aspects of the text that are important to the task at hand.

Here we will provide you with a recipe of commonly-used tools for simplifying texts to represent them quantitatively. We stress that there is not a one-size fits all solution to preprocessing texts. The steps you take to represent the text will necessarily depend on your specific research question and focus. Indeed, the choices you make about how to represent text will affect your subsequent analyses, so these choices should be made deliberately and with care (Denny and Spirling, 2018). If you think you might want to revisit these choices as your research question becomes more defined, we suggest that you set aside a subset of your data so that you can validate your inferences at later stages of the research process, or have a plan for collecting more data that you can use for final inferences.

The first step to transforming your text into data will be to digitize the text if it is not digitized already. While text collected from social media and the Internet will already be machine readable, documents from archives might be handwritten or printed on paper. In order for a computer to recognize the words, researchers can use *optical character recognition*, technology that converts pictures of text into text that a machine can read. These technologies are trained by corresponding known pictures of words with the machine-encoded text; the algorithm can then predict the appropriate machine encoded character based on a new picture. While not perfect, these technologies are the best solution for making text machine readable for large corpuses where transcribing the documents by hand would be prohibitively time consuming.

Once you've digitized your text, the first decision you must make is what is the *unit* for your analysis, what is the basic unit of text that you want to describe? In many cases, the unit of analysis is the document – most researchers want to categorize documents from newspaper articles to social media posts to journal articles. However, in some cases the unit of analysis might be smaller or larger than the document – the analyst may want to break apart the document in order to categorize sections, paragraphs, or sentences or may want to describe a collection of documents such as journals, a newspaper, or a book. Whatever text you decide to use as your basic unit of analysis, the collection of these units is called a *corpus*, or the full dataset on which you will run your analyses.

The purpose of the analysis is to describe the units of text, for example, to put each into specific categories or to summarize the main themes of all of the units in the corpus. To do this quantitatively, each unit of text must be turned into data, in which the text is encoded into *features*. The features of the text are any aspect that can be measured on all of the texts. Features are commonly described in data analysis as variables. One simple example of a feature that could be included in the dataset is the number of times a particular word appears within the unit; for example a count of how many times each text unit contains the word "political." In addition, features can be metadata about the text – the time period the text was written in, the author of the texts, or the newspaper the text was printed in for example. Features can also be a derivation from the words; for example, whether or not the text contains a proper name or in other cases a low-dimensional projection of the words. We will describe each of these types of features in this chapter in turn.

## 3.1 Tokenization

The basic problem of representing the text is deciding which features to use to describe your text units in your analysis. As we described in the last chapter, texts are *high-dimensional* in that there are many possible features that could be extracted from of any given unit text. Texts not only contain information about what words are in them, but also the order in which these words appear, and other metadata associated with the document. Typically, there is too much information within the text and all of it cannot be represented efficiently by the data.

To simplify the transformation of text into data, many methods for text analysis use what is known as the "bag of words" assumption, the idea that features should include counts of unique words in the corpus, often called "unigrams," which are separated by spaces. With the bag of words assumption, each individual word is a feature or token, and the number of each of the words in the text is used to describe the text, but nothing is retained about word order.

The "bag of words" assumption is a useful starting place for developing a feature set for texts, but is not ideal for all contexts. In many languages, for example Chinese, Japanese, and Lao, words are not separated by spaces; instead, words are inferred by the reader from the context within the sentence. In these languages, the analyst first has to identify the words that should be included as individual features, a process commonly known as *tokenization*. For each of these language, software has been developed to tokenize the words, or separate them by spaces. The bag of words methods can then be applied directly.

In other cases, "unigrams" or individual words discard important word order that might be important to retain. Some concepts bridge words, a common example is "White House," where the meaning is lost when the phrase is tokenized into "white" and "house." For some applications, word order may be particularly useful in predicting the categories of the documents or describing the main themes of the corpus. In this case, the analyst can retain "n-grams," or tokens that include $n$ words. For example, the researcher may retain all "bigrams," or consecutive sets of two words, or "tri-grams," consecutive sets of three words. In this case, each unique bigram or trigram is a feature, and the number of each bigram or trigram is used to describe each unit of text.

The researcher can include n-grams in two ways. First, she could include all consecutive sets of two words that appear in the corpus, in addition to or instead of unigrams. This might be most useful in cases where she performs another step of automated feature selection, where the computer decides what of a given set of features to retain, something we will discuss later in the book. Alternatively, the researcher might find it useful to retain only certain bigrams and trigrams that she anticipates will be useful to her analysis. For example, she might know that "White House" or "International Relations" often appear together and therefore may specify which bigrams should be included as one token. A variety of approaches exist to suggest bigrams and trigrams that frequently appear together for inclusion within a corpus (Handler et al., 2016).

For now, think of tokenization as a process that either inserts spaces in between words (such as in the case of Chinese, Japanese or Lao texts) or removes spaces in between words to incorporate n-grams. Each space indicates that information about the word before or after that space is completely discarded, the only information retained are the words within the spaces. Below, we show an example of how a document might be tokenized in a Chinese document and in an English document where the researcher chooses to include selected, but not all, bigrams.

**Document 1**

"人生就像蒲公英，看似自由，却身不由己。有些事，不是不在意，而是在意了又能怎样。"

" 人生 就 像 蒲公英 ， 看似 自由 ， 却 身不由己 。 有些 事 ， 不 是 不在意 ， 而 是 在意 了 又 能 怎样 。 "

**Document 2**

"Climate change constitutes a serious threat to global security, an immediate risk to our national security ... And so we need to act – and we need to act now. –President Obama"

"Climatechange constitutes a serious threat to global security, an immediate risk to our national security ... And so we need to act – and we need to act now. –President Obama"

## 3.2 Discarding Complexity in Word Features

Once the words in the document have been tokenized, analysts often discard features that are unimportant to the analysis. For example, if the researcher is not interested in how many commas, apostrophes, or periods appear within the text, then choosing to delete all punctuation might be useful since it will decrease the total number of features in the data. Second, the analyst may decide to remove *stopwords* or common words used across documents that do not give much information about the categories the researcher wants to place the texts in. In English, common words such as “the,” “that,” and “and” might be removed from the documents to reduce the size and complexity of the text. All languages have such commonly used words that function as prepositions or articles. Lists of stopwords in many languages are available in packages that provide text analysis tools, and statistical software programs for text analysis often have automated functions to remove punctuation and stopwords.

The researcher may also want to get rid of formatting cues that often exist within data from other sources. For example, section headings, page numbers and html tags, may not be important to the purpose of the research, and, if so, should be discarded before the analysis.

Researchers may combine some of the features that describe the text by removing the endings of words, or reducing words to their root. Those working with texts in the English language typically discard the ending of conjugated verbs and plural nouns to merge similar words into one root, in a process called “stemming.” For example, “family,” “families,” “family’s” and “familiar” would all be replaced with “famili” in the text. This stemming process reduces the number of features that the analyst has to deal with, while not discarding significant information since words that begin with famili all have similar meanings. There are many stemming algorithms available in a wide range of software; among the most common is the *Porter* stemming algorithm (Porter, 1980).

A more complex version of stemming is lemmatization, which rather than simply removing word endings, reduces words to their “lemma,” or base root. For example, “see,” “saw,” and “seeing” would not be reduced to one simplified word with stemming, but with lemmatization these words will all reduce to the word “see.” Lemmatization is typically slower than stemming because the algorithm considers the word in its context (Grimmer and Stewart, 2013; Jursfsky and Martin, 2009; Manning, Raghavan and Schütze, 2008). However, in languages such as German, where conjugations are not always represented by adding an ending to a word, lemmatization can be more useful than stemming.

Below we provide Document 2 from above, but where the text has been converted to lowercase, tokenized, removed of punctuation and stopwords, and stemmed, in that order. As of yet, the researcher has already made important decisions about which features he will ultimately include within the analysis. Note that the order in which these steps are taken matters as well. For example, if stopwords are removed before tokenization, the analyst will end up with different bigrams and trigrams than if they are removed before. Researchers should examine the texts after pre-processing to ensure that the result of pre-processing will make sense for their analysis.

When pre-processing steps the research wants to do are not available in software, researchers can use *regular expressions* to manipulate text to customize preprocessing to their own dataset. Regular expressions may also be useful for discarding unnecessary text such as formatting or html that interrupts the text and is unnecessary to the analysis. Regular expressions are a language for searching general string patterns. They form the basis for string search for many programming languages, search engines, and text editors. They also provide functionality for replacing or removing unwanted text. For more information on how to use regular expressions, along with examples of regular expressions see Jursfsky and Martin (2009, Chapter 2).

**Document 2**

"Climate change constitutes a serious threat to global security, an immediate risk to our national security ... And so we need to act – and we need to act now. –President Obama"

"Climatechange constitutes a serious threat to global security, an immediate risk to our national security ... And so we need to act – and we need to act now. –President Obama"

"climatechang constitut serious threat global secur immedi risk national secur need act need act now presid obama"

## 3.3 The Document-Feature Matrix

Once you have tokenized the corpus, removed unnecessary words, word endings, and formatting, you are ready to turn text into numeric data. We will use $N$ to delineate the number of unique text units within your dataset, and $i$ to index each of these units. After preprocessing these units, you can identify all unique tokens within your dataset – the total number of unique tokens within your dataset we will delineate by $J$ for the rest of the book. We will call this the "vocabulary." Each unique feature, indexed by $j$, becomes a variable within your dataset, and the the number of times each unique token appears, $w_{ij}$ within the text becomes your data. In other words, for each unit of text, you will count how many times each token is used within that text and record that as your data.

The result of this is what we call a document-feature matrix, which we will call $\boldsymbol{W}$, a $N \times J$ matrix. In this matrix, there are $N$ rows that total the number of unique units of text, and $J$ columns that total the number of unique features. Each element of this matrix $w_{ij}$ is the number of times that unique token appears in each unit of text. In general, since most documents only contain a few of the many unique tokens within a corpus, the document-feature matrix is relatively *sparse*, or contains lots of zeros.

For example, say that we have the three documents in a corpus of Presidential quotes, below:

> **Presidential Quotes Corpus**
>
> “Climate change constitutes a serious threat to global security, an immediate risk to our national security ... And so we need to act – and we need to act now. –President Obama”
>
> “Whenever America fights for the security of our country, we also fight for the values of our country.–President Bush”
>
> “Climate change is more remote than terror but a more profound threat to the future of the children and the grandchildren and the great grandchildren – President Clinton”

We could then put them through the same preprocessing routine as we did with Document 2, above. This would create three documents, each having been converted to lowercase, punctuation removed, stopwords removed, stemmed, and converted “climate change” into a n-gram “climatechange”. After doing this, we would document the number of unique words within the corpus, the *vocabulary.* Overall, after being preprocessed these three documents have 29 unique words. $\boldsymbol{W}$ would therefore consist of three rows (one for each document) and 29 columns (one for each unique word). The entries of those rows would count the number of times each word appeared in each document, see below. See the document-feature matrix for the Presidential quotes corpus below.

| | climatechang | constitut | serious | threat | global | secur | immedi | risk | nation | need | act | now | presid | obama | whenev | america | fight | countri | also | valu | bush | remot | terror | profound | futur | children | grandchildren | great | clinton |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| text1 | 1 | 1 | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 2 | 2 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| text2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 2 | 2 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| text3 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 2 | 1 | 1 |

Table 1: Document-feature matrix $\boldsymbol{W}$ from the Presidential quotes corpus.

Once we have $\boldsymbol{W}$, there are many operations we can perform on it that can help summarize the text. To take two simple examples to start, if we sum each of the columns of $\boldsymbol{W}$, we will have a summary of how many of each unique token appear within the corpus. This could then be inputted into a word cloud or a table to understand the most common words within the corpus.

| | climatechang | constitut | serious | threat | global | secur | immedi | risk | nation | need | act | now | presid | obama | whenev | america | fight | countri | also | valu | bush | remot | terror | profound | futur | children | grandchildren | great | clinton |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 1 | 1 | 2 | 1 | 3 | 1 | 1 | 1 | 2 | 2 | 1 | 3 | 1 | 1 | 1 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 1 | 1 |

Table 2: Result of summing the columns of $\boldsymbol{W}$ , the document feature matrix.

| | |
|---|---|
| text1 | 17 |
| text2 | 11 |
| text3 | 12 |

Table 3: Result of summing the rows of $\boldsymbol{W}$, the document feature matrix.

Or, we could sum each of the rows of the document-feature matrix, which would tell us how many unique tokens each document has. The maximum of this sum would be the length of the longest document, and the minimum of this sum would be the length of the shortest document. While these are basic examples, much of the rest of this book describes algorithms that perform operations on this document-feature matrix.

### 3.3.1 Document-feature matrix examples

Already, the simple creation of the document-feature matrix facilitates analyses that allow us to better understand the content of text and make inferences about the social world. Most simply, as we showed above, taking the column sums of $\boldsymbol{W}$ allows us to create histograms of the word counts within a corpus. If we then select the $N$ largest column sums, we can

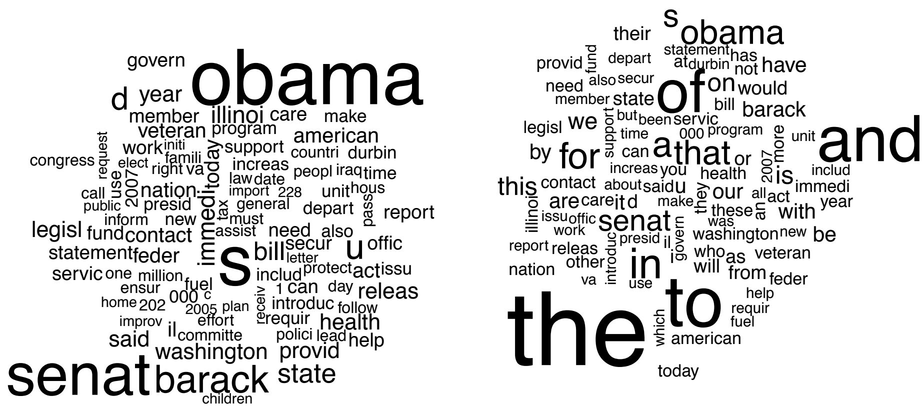


Figure 1: Word Clouds of Obama Corpus. Left: stopwords removed. Right: no stopwords removed.

quickly understand the most central tendencies of the text. Often, these most frequent words within a corpus are summarized by a word cloud, where the size of the word corresponds to the relative frequency of that word in the corpus.

Although wordclouds have been criticized for their simplicity, you can think of them as just a histogram of the most frequent words, more efficiently presented to the reader. The simple example of a wordcloud demonstrates how preprocessing can impact the inferences we can draw from the data. Figure 1 shows two wordclouds: both from a dataset of Barack Obama's press releases when he was serving in the Senate – but one with removing stopwords and one without having removed stopwords. Clearly, the wordcloud without stopword removal is less informative about the primary topical content of the releases to the reader than the one with stopwords removed.

In other cases, stopwords are of primary interest to the researcher. For example, Mosteller and Wallace (1963) explicitly use stopwords in their analysis to identify the authorship of text. Their focus is the Federalist Papers, 77 documents written by a combination of Alexander Hamilton, John Jay, and James Madison in the late 1700s supporting the American Constitution. For twelve of these documents, it was unknown whether they were written by Hamilton or Madison. For three documents, it was known that they were written jointly, but which author wrote which section was disputed.

Mosteller and Wallace (1963) make the observation that stopwords (what they call *filler words*) distinguish the two authors – Hamilton and Madison use words like *upon*, *by*, and *to* at different rates in documents known to be written by the two authors. The decide on a unit of analysis as blocks of text of about 200 words each, in order to be able to distinguish joint authorship within one document. They then count the use of filler words in each of these blocks, similar to our creation of a document-feature matrix.

The authors then create a model that uses the stop words to predict the authorship of papers using existing known documents written by the two authors. They then predict the authorship of the unknown papers using the estimated relationship between stopwords and authorship. The model distinguishes authorship in the unknown documents quite decisively – attributing all 12 documents with unknown authorship to Hamilton.

The contrast in the features of interest between Mosteller and Wallace (1963) and the word cloud illustrate how the best pre-processing steps will depend on the ultimate quantity of interest the author is interested in measuring. Pre-processing is not a step to be overlooked or undervalued in the analysis and may be consequential for the results of an analysis. Denny and Spirling (2018) show that variations in the steps of pre-processing can impact the results of the analysis and suggest testing several different combinations of pre-processing to see if the outcome of the analysis differs. Further, authors such as Schofield and Mimno (2016); Schofield, Magnusson and Mimno (2017) have shown that for certain measures of quality of topic models, pre-processing steps such as stemming and removal of stopwords may not confer advantages and may even be detrimental to the overall analysis.

Our advice is that researchers should approach pre-processing differently at different stages of the research process. We think that what we have presented here are good guidelines for getting started with pre-processing, before jumping in and looking at a corpus. But pre-processing should be returned to throughout the research process. At the stage of discovery, which we describe in the next chapter, different methods of preprocessing will lead to different compressions that may reveal insights that are useful in their own right, even if the analyst did not anticipate them. For example, a researcher might fail to remove stopwords in one analysis and realize that the use of the words "he" and "she" are relevant to understanding important patterns in the text. In the discovery phase, the fact that you get different answers from pre-processing the corpus isn't necessarily troubling and can even be illuminating – there is no "best" method of pre-processing because you are still uncovering the research question.

Once we have a clear goal in the analysis, when we are ready to measure a concept of interest, the question of the "best" method of pre-processing becomes more tractable. While the analyst may have a good intuition about the kinds of words that they should retain that would lead to the most accurate measure for a particular concept, preprocessing at this stage of the research process is fundamentally an empirical question. When we know the concept we are interested in, and even have a sample of hand-coded documents that we think show this mapping, we can revisit preprocessing to determine what combination of words leads to the most accurate representation of the compression we are trying to resolve from the text.

Because each analysis will require a different measure, or a different compression of the text into categories or quantities of interest, we do not believe there is a one-size fits all approach to pre-processing. We suggest that at the early stages in the research process, researchers think carefully about the ultimate quantity of interest that they might want to measure and take pre-processing steps they think will be in service of that measurement. However, because the right pre-processing steps to take may not always be obvious ex-ante, researchers should be aware that they will likely have to revisit these procedures in the discovery and measurement stages, which we describe in later chapters.

## 3.4 Non-textual features

In addition to using the words as the corpus as features that describe a unit of text, we will often use features outside of words within the text as important parts of the analysis. Documents not only have words that describe them, but have other information about the context in which they were written. There might be other, non-textual features that play an important role in communication that we might want to add to the corpus. For example, we might have the text of the words within a news broadcast, but want to add information about the tone of the speaker, or the music playing in the background (Schonhardt-Bailey, 2017). The date or time period it was written, the book or newspaper the article appeared in, the author of the document, or other attributes of the author such as gender, age, or political party, might be important information to add to the feature set. These attributes can be used in conjunction with the text to classify the documents, or might be used as primary inputs in the analysis. We will denote this $N \times P$ matrix of non-textual features throughout this book as $\boldsymbol{X}$, an additional matrix to $\boldsymbol{W}$ that describes the documents of interest.

## 3.5 Leveraging text complexity

So far, we have described very basic processes of turning text into data: first tokenizing the document using n-grams, next removing unnecessary complexity, and creating a document-term matrix. For the most part, our assumption so far has been that the order and context of the words is not important – we have been using the "bag of words" assumption to simplify the text.

While as we will discuss later, this simple approach is surprisingly effective for many research questions, it may be too simple for some tasks. One of the problems with the bag of words approach is that words can be ambiguous and imprecise, causing confusion because words are taken out of context. For example, one feature of the document-feature matrix might be "crack." Does this refer to a break, as in a crack in a window pane, "cracking" or solving a problem, or the nickname for cocaine? Two words might also have the same meaning, but have two separate features. For example, "Dairy Queen" could also be called "DQ" and similarly "McDonald's" "Mickey D's." While these difference could be unimportant for some problems, their separation might be problematic for others.

In this section, we will describe how to add information about the order and word context to the document-feature matrix $\boldsymbol{W}$, sometimes called "resolving" ambiguities within the data. Their resolution can be used to add information to the document-feature matrix by adding information about context, or simplifying$\boldsymbol{W}$ by combining two separate features that should be considered one feature.

### 3.5.1 Parts of Speech Tagging

Tagging the parts of speech within a sentence can be useful for resolving ambiguities in the data, either by combining words that have the same parts of speech, or disambiguating words that are the same by have different meanings depending on their parts of speech. Parts of speech tagging assigns a part of speech to each word within a document. For example, crack could be tagged with "verb" in "I cracked with window" or a noun in "The window has a crack." Tagging of each word is either done by a rule-based algorithm, but more often is done by machine learning algorithms that use an already tagged corpus to learn tags for each word based on their relationship to each other (Jursfsky and Martin, 2009, pg 297). For example, the Penn Treebank is a corpus of 4.8 million words from a variety of English language sources tagged with 45 different parts of speech tags and is often used as a training set for parts of speech tagging in English (Marcus, Marcinkiewicz and Santorini, 1993)

As a simple example, we tag the parts of speech of the quote from President Bush from above, shown below. The parts of speech tagger finds four words as nouns – "security," "country," "values," and "country" – and three words as proper nouns – "America," "President," and "Bush." Note that because of the context of the document, the parts of speech tagger is able to distinguish from the plant "bush" from the last name "Bush."

> Whenever ADV America PROPN fights VERB for ADP the DET security NOUN of ADP our ADJ country NOUN , PUNCT we PRON also ADV fight VERB for ADP the DET values NOUN of ADP our ADJ country NOUN .– PUNCT President PROPN Bush PROPN

Parts of speech tagging can be useful in augmenting the feature set. For example, while the bag of words approach would merge both the noun and verb versions of "crack" into one feature, after identifying the parts of speech, the research could include one feature for crack-verb and one feature for crack-noun. Researchers may also be interested in only using particular types of speech for an analysis or components of an analysis. For example, O'Connor, Stewart and Smith (2013) develop a model to characterize the topics of events that happen between a particular object and subject within a sentence. For this, they want to characterize the verb in between a selection of objects and subjects in the text. In this case, parts-of-speech tagging allowed them to specify that particular parts of the topic model should only consider verbs within the corpus.

### 3.5.2 WordNet

Parts of speech tagging facilitates larger groupings of words into groups or common themes. For example, researchers have developed large databases that document relationships between words that have the same part of speech in the English language to facilitate linking similar words together. The largest of such English language project is WordNet (Miller et al., 1990). WordNet allows researchers to query a database of nouns, verbs, or adjectives to retrieve information about a given word. The information returned includes definitions and synonyms as well as relationships with other words; for example, allowing researchers to extract more general concepts that subsume a word in the text and others (for some good examples, see Jursfsky and Martin (2009, pg 603)).

WordNet can allow for the simplification of the document-feature matrix by identifying hierarchies of words. For example, the researcher might only be interested in whether a word indicated that the noun was a car, and therefore would want convertible, truck, and car to be merged into one feature "car." Some algorithms focused on measuring the readability of text use WordNet to identify the number of causal verbs and causal predicates, an indicator of how complicated the text is (Graesser et al., 2004). Other researchers have used WordNet to help human coders identify a set of emotional words in text, finding synonyms for emotions such as anger, disgust, fear, or joy (Strapparava and Mihalcea, 2008).

### 3.5.3 Dependency Parsing

Parts of speech tagging in turn is an input in a technique called dependency parsing, where each word is tagged with its relationship to other words within a sentence (Jursfsky and Martin, 2009, Chapter 14). Such a system is called dependency parsing because it describes a sentence in terms of the ways in which each word in the sentence depends on other words. It identifies a root word (often the primary verb) and then describes the other words in relation to that word and to each other. For example, in the sentence, "I read a great book," the subject of the sentence "I" and main verb of the sentence is "read," "read" in this case would be identified as the root of the sentence. The subject of the root is "I" and the object of the root "read" is "book"; both "I" and "book" depend on "read." The sentence can then be represented as a tree, as shown in Figure 2. The adjective "great" modifies the object "book". A project called "Universal Dependencies" (Nivre et al., 2016) has created a system of dependencies that is used in analyses of text.

Dependency parsing has been frequently used in the extraction of events from text, what

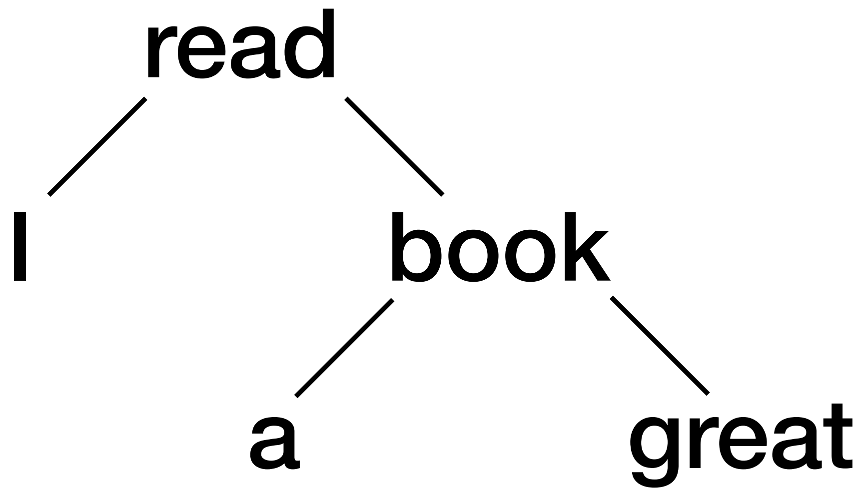


Figure 2: Example parse tree for the sentence "I read a great book."

is often know in social science as "event data." We describe event data more in Chapter 5, but it attempts to capture actions of the form 'someone does something to someone else' possibly located in a particular time and place. Because the events to be extracted are directed, the underlying structure of the sentence is incredibly important 'A attacks B' is very different from 'B attacks A.' Early approaches captured this directional information using word order, but this was complicated by passive voice 'B was attacked by A.' Dependency parses identify the grammatical relations in the sentence which allow us to code events straight from the tree rather than from the word order in the sentence (Beieler, 2016). Dependency parse representations can even be combined with discovery methods of the sort described in the next chapter to discover types of events. O'Connor, Stewart and Smith (2013) use this strategy to learn types of events between countries and non-state actors inductively from news reports.

### 3.5.4 Named-Entity Recognition

Last, parts of speech tagging can facilitate the recognition of people and places within text. Using similar methods of classification from tagged texts as parts of speech tagging, algorithms have been developed to tag proper nouns within text (Manning et al., 2014). This process, called named-entity recognition (NER) can also be used to simplify or complicate the document-feature matrix. For example, two people with the same name might be distinguished by where they live or the places that they are associated with. Anne Hathaway may mean a different person in the context of William Shakespeare (wife of William Shakespeare), than in the context of *The Devil Wears Prada* (American actress). Other names mentioned within the document could therefore be used to separate two individuals in different contexts. NER could also be used to reduce the dimensionality of the document-feature matrix. If an analyst is only interested in people mentioned within the documents in order to form a network of co-occurrences between entities within the documents, after running NER they could subset the vocabulary to only terms that are recognized as proper nouns. The network of co-occurrences is then defined as the $\boldsymbol{W}^T\boldsymbol{W}$, where $\boldsymbol{W}$ is the document-feature matrix.

For a simple, example of named-entity recognition, we used NER from the Spacy Python package to identify named entities within the quote from President Bush. Note that not only does the algorithm recognize the entities within the quote, it also identifies the type of entity – in this case that "America" is a geo-political entity and that "Bush" is a person. It also records the location of the entity in number of characters from the beginning of the document.

```
('America', 9, 16, 'GPE')
('Bush', 113, 117, 'PERSON')
```

Jaros and Pan (Forthcoming) use named-entity recognition software in official Chinese newspapers to better understand the distribution of power among political elites in China. Many had speculated that after he took office in November 2012, General Secretary Xi Jinping consolidated power more than his predecessor, Hu Jintao. Other China watchers observed that the Party had consolidated more power relative to the government in China and that foreign influence seemed to be declining in China.

Of course, all of these observations are based on impressions, and difficult to validate with data. However, in China official newspapers typically reflect Party positions. To measure power, Jaros and Pan (Forthcoming) used named entity recognition in Chinese official newspapers to recognize all people and organization named in Party newspapers. They measure power by using the number of times an entity was mentioned within official newspapers, as a proportion of all other entities mentioned. They found – consistent with speculation – that the official newspapers reflected many of these trends. Xi Jinping was much more mentioned in Party newspapers than his predecessor was during his time in office. Party institutions also gained prominence over the time that Xi was in office. However, they show unexpectedly that there was not a decline in mentions of foreign organizations.

## 3.6 Vector representations of documents and similarity metrics

Once $\boldsymbol{W}$ has been created, information about the documents and their relationships to one another become easy to extract. Each document is now represented by a vector of numbers, which signify the amount or number of each of the features it includes. Mathematical operations are now easy to perform on this newly-constructed matrix with documents represented as vectors.

Most simply, similarity metrics can be computed on the document set to measure the similarity between any pair of documents. There a variety of similarity metrics to choose from, each with its own notion of similarity or distance. One simple way to measure similarity between documents might be compute the inner product between two documents. This would involve multiplying each corresponding entry in two document vectors together and then summing these terms together. The more non-zero terms in the two vectors overlap, the larger the resulting number produced is, indicating higher levels of similarity.

However, one problem with the inner product is that its result depends on the length of the two documents. If I duplicated all words so they were written twice as frequently in one document, my inner product would increase even though the similarity between the two documents has arguably remained unchanged. One alternative to using the inner product is to instead use the cosine of the angle between the two documents as a measure of similarity, called *cosine similarity*.

Cosine similarity is more simple than it sounds. Imagine you had only two words within your corpus (shown in Figure 3): "computer" and "geek." Say the first document included four of the word "computer" and one of the word "geek." Then the second document had two of the word "geek" and one of the word "computer." You could draw the feature space of these two very simple documents in two-dimensions. The angle between these documents is a measure of similarity – if the angle is smaller, then the documents are more similar. By taking the cosine of the angle, we normalize the angle – which will range from zero to 90 degrees – to a scale from zero to one – one being most similar and zero being most different. Of course, the corpuses we will work with will rarely ever have only two words, so we have to imagine the angle between the two documents vectors in much higher dimensions than just two. But the intuition is the same – we are looking for the similarity of the two documents by measuring the angle between their projection onto the feature space.

Cosine similarity can be computed simply by standardizing the inner product. First, take each word within each document and divide it by the length of the document. Then, take the inner product of the two resulting vectors.

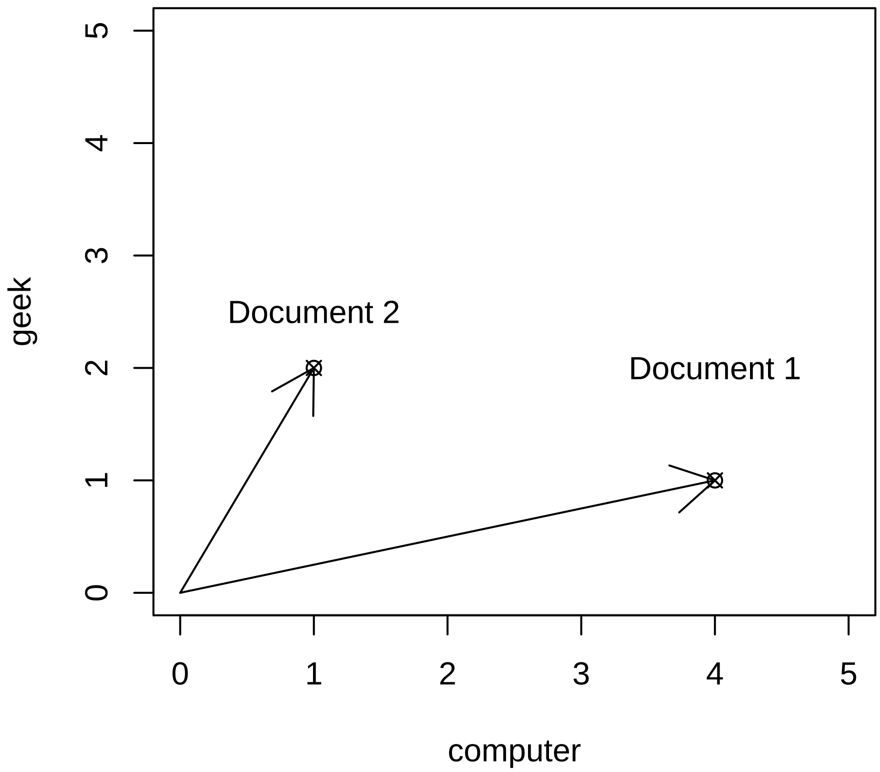


Figure 3:

$$cos\theta = \left(\frac{Doc1}{||Doc1||}\right) \cdot \left(\frac{Doc2}{||Doc2||}\right)$$

where $||Doc1||$ is the length of the first document. Note that cosine similarity does not change if you simply duplicate the words within one document since the angle between the two vectors will stay the same.

For example, say we are interested in a corpus of President Barack Obama's press releases from his time in the Senate, from 2005-2008 (Grimmer, 2013). While reading through the documents, we became interested in one particular press release about taxes, entitled "Obama Calls on IRS to Protect Taxpayers Privacy." We'd like to know whether or not there were more documents in the corpus like this one.

One way to go about this would be to read all of the 709 documents in the corpus. But another way would be to use the methods of text as data you already know how to use since you've made it this far in the chapter. First, we would remove punctuation, convert to lowercase, remove stopwords, stem, and do whatever other more complicated pre-processing techniques we think might capture the features that will reflect the similarities we are looking for. Then we would convert the 709 documents into a document-feature matrix, $\boldsymbol{W}$. With a few lines of code, we could calculate the cosine similarity between all other 708 documents and the document we are interested in. We would find that documents with the highest cosine similarity to "Obama Calls on IRS to Protect Taxpayers Privacy" include "Obama Introduces Bill to Stop Tax Preparers from Selling Confidential Tax Information," "Obama Calls on the TSA to Address Recent Security Vulnerabilities at O'Hare," and "Obama and Coburn Introduce Strengthening Transparency and Accountability in Federal Spending Act of 2008," all releases that have to do with privacy and/or taxes.

Of course, there are many other metrics and notions of similarity and distance that produce different similarity metrics than cosine similarity. A researcher could also measure distance by simply taking the Euclidean distance between two documents, the square root of the sum of the squared differences between each of the features:

$$= \sqrt{\sum_{j=1}^{J}(w_{1j} - w_{2j})^2} \tag{3.1}$$

On the Obama dataset, if we take the Euclidean distance between the document of interest and all of the 708 other documents, the document with the smallest difference to "Obama Calls on IRS to Protect Taxpayers Privacy" is "Statement of Senator Barack Obama on the President's Signing of the Genetic Information Nondiscrimination Act." While this document also deals with privacy, Euclidean distance provides a different answer to what document is closest than cosine similarity, illustrating how the selection of the distance metric will have consequences for the answer provided by the analysis.

Euclidean distance is often called the $L^2$ distance or the L2 norm. It corresponds to the direct distance between two points in space. Alternatively, we could use the $L^1$ distance, otherwise known as the L1 norm or "Manhattan distance." The idea behind Manhattan distance is that it measures the distance between two points in space, when you can only travel in right angles. Euclidean distance is therefore the distance as the crow flies, Manhattan distance is the distance as a taxicab would drive in Manhattan, which is why it is also commonly referred to as the "taxi cab metric." Manhattan distance is taken by the sum of absolute differences between each of the features:

$$= \sum_{j=1}^{J} |w_{1j} - w_{2j}| \tag{3.2}$$

In the Obama dataset, if we find the document with the minimum Manhattan distance, the closest document is "Obama Statement on the Tragedy at Virginia Tech." This appears to be less relevant than the documents produced by Euclidean distance and cosine similarity, perhaps an indication that at least in this case the Manhattan metric corresponds less to our sense of similarity than the other two.

Because the outcome of string similarity analysis is a $NxN$ matrix that describes the similarity between each of $N$ documents, string similarity metrics can also simplify a more complex features space, allowing the researcher to take advantage of more of the word order of two strings. *String kernels* (Lodhi et al., 2002) are more complicated string similarity metrics that compute similarity based on the overlap of ordered characters between two strings. For example, "card" and "carefully" both have overlapping character strings of size three, "car".

In their most simple form, string kernels map out all sets of $n$ consecutive characters within two strings. To compute similarity, they take the inner product of these sets of $n$ consecutive characters between the two strings, and then normalize to account for string length. For example, take the features of two documents from the term document matrix $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$. These vectors can be mapped into a larger feature space that takes each unique $n$ consecutive sets of characters. For example, the word "barn" maps to "ba", "ar", and "'rn" when $n = 2$. If we do this mapping for both vectors, we get another two vectors of larger dimension $V_1$. The simplest way to calculate a kernel is then to take the inner product of these two vectors:

$$K(\boldsymbol{w}_1, \boldsymbol{w}_2) = \langle \phi(\boldsymbol{w}_1), \phi(\boldsymbol{w}_2) \rangle \tag{3.3}$$

where $\phi$ is the transformation from the original to larger feature space. Typically, this inner product is then normalized. The closer this normalized number is to 1, the more similar the two strings are. Some methods described later in this book are a type of method called "kernel methods," which means they can be computed by the $NxN$ kernel matrix and don't require extraction of all of the features in the corpus. This allows them to be efficiently computed even with complicated feature sets. Spirling (2012) uses one of these approaches – Kernel Principal Components Analysis – to scale treaties between Native Americans and the U.S. government. In the Obama dataset, documents with the high string kernel similarity to our document of interest include"Obama Calls on the TSA to Address Recent Security Vulnerabilities at O'Hare" and "Obama Calls on Labor Department to Prevent Minimum Wage and Overtime Violations."

## 3.7 Vector Representations of Words

So far, we have only represented text as a bag of words, or by whether or by vector of features that indicate the number of each word included within a document. Outside of introducing tools such as WordNet and Named Entity Recognition, we haven't used any information about the words' relationships with each other or the order in which they appear within the document. When we compute similarities between documents, there are no gradations of how similar words are across documents. For example, if "bus," "car," and "beautiful" are all unique words within the vocabulary, "bus" and "car" are equally similar in the bag of words context as "bus" and "beautiful." But of course, we know much more about any given word – we know the context in which it was used within the document and other words within the language that we are working in that typically appear around it. As we explained before, we could include n-grams within the document-feature matrix to capture some of this context. But in a large corpus, this list of possible n-grams becomes very long and computationally burdensome. Indeed, it may not be important to document every other word a word has ever been used with, but rather to capture something more general about each word and its similarities to other words.

### 3.7.1 Word Embeddings

One way to address this problem is to use word embeddings to encode more information about individual words. The idea behind word embeddings is to project each word in the vocabulary onto an n-dimensional space with all other words. That is, instead of representing words as a binary, 0/1 variable that indicates its presence or absence, instead represent each individual word as a vector with length $K$. Words that are nearer to each other within this $K$-dimensional space will be more similar to each other – more likely to be substitutes or synonyms for one another.

Note that this is a much more information-rich way of describing text – instead of describing documents as vectors, we are now describing words, which is a more granular level of analysis – as vectors. The problem with this approach is that typically the unit of analysis of interest to the research is at the *document* level. To address this, the word embeddings can be computed at or aggregated to the document level in order to describe the area of the word space that the document exists in. Documents that are in similar spaces are more likely to be similar to each other. The advantage of this approach is even if the documents do not use the exact same wording, if they use words that are commonly used as substitutes for one another, they will still occupy a similar space.

Word embeddings can be computed in many different ways – from computing word co-occurrence statistics, to computing Principal Components Analysis on the words themselves and Latent Semantic Analysis. All of these methods output a matrix with the number of rows represent each unique word in the vocabulary an each column is a representation of where the word lies within the space of all vocabulary (Bengio et al., 2006). For purposes of this section, we will go through one increasingly popular method of creating word embeddings, called the skip-gram model (Mikolov et al., 2013). In the skip-gram model, word vectors are computed by setting up a prediction problem where the words surrounding a selected word predict the selected word as if it was missing. For example, "the quick brown fox [blank] over the lazy dog" would use the estimated word vectors of "the, quick, brown, fox, over, the, lazy, dog" to predict the word in the blank. The idea is that words that are used in similar contexts will have similar word vectors; they will predict similar words to fill in the blank. By iterating this window across all words in the corpus, the objective is to create a set of word vectors that maximize the average of the probability of seeing the correct word $w_t$ given the words within the context $w_{t-k}, ..., w_{t+k}$, following Le and Mikolov (2014):

$$\frac{1}{T} \sum_{t=k}^{T-k} \log p(w_t | w_{t-k}, ..., w_{t+k})$$

where $p(w_t | w_{t-k}, ..., w_{t+k})$ is parameterized as:

$$p(w_t | w_{t-k}, ..., w_{t+k}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_{w_i}}}$$

where $y_{w_i}$ is the unnormalized probability of seeing any word $w_i$ given the words in the context $w_{t-k}, ..., w_{t+k}$, given by

$$y = b + Uh(w_{t-k}, ..., w_{t+k}; \mu)$$

Here $U$ and $b$ are coefficients that are optimized, along with $\boldsymbol{\mu}$ which is a matrix where each column contains the word vector for each unique word. $h$ is a function that averages the word vectors for $w_{t-k}, ..., w_{t+k}$ that are selected from $\boldsymbol{W}$. Mikolov et al. (2013) also provide a relatively fast implementation to train word vectors on large corpuses called word2vec.

The output $\boldsymbol{\mu}$ of word2vec allows us to describe similarities among words within the corpus. We can use cosine similarity as described above to examine similarities between words. For example, we trained a word2vec model on all unigrams and bigrams from a corpus of President Barack Obama's press releases from his time in the Senate, from 2005-2008. We then looked for the most words closest in the word2vec space to "climate_change" and "pollution." Figure 4 plots the most similar words in terms of cosine similarity in the word2vec space to these two words. As you can see, some words are used very similar to both climate change and pollution – words such as "emissions" and "greenhouse gas" are highly similar to both. But then there are words that are more likely to be similar to climate change and not pollution – particularly words about international affairs like "our dependence" and "foreign oil." Similarly, there are words that are more likely to be used as substitutes for pollution, such as "clean water" or "great lakes" that are less likely to be used as substitutes for climate change. This analysis sheds some insight into the ways in which Obama used the word climate change – in a global context – and alternatively the way he used the word pollution – in a local, Illinois-specific, context.

While word vectors produced by word2vec are typically used as inputs in classification and translation problems, they also have a number of intuitive properties that make them interesting in their own right. Not only do similar words appear in similar places in the space, analogous words appear in parallel planes of the space. This means that arithmetic can be done on the word vectors themselves – in one of the corpuses trained by Mikolov et al. (2013) subtracting the word vector for "man" from the word vector for "king" gives you the address close to the word vector for "queen."

In the Obama corpus, we can use this arithmetic to understand how Obama used particular words in different contexts during his time in the Senate. For example, the closest words to the vector that results from adding the word vector for "poor" and the word vector for "tax" is "ssi_eligibility," "gross_income," and "retirement_plans." On the other hand, among

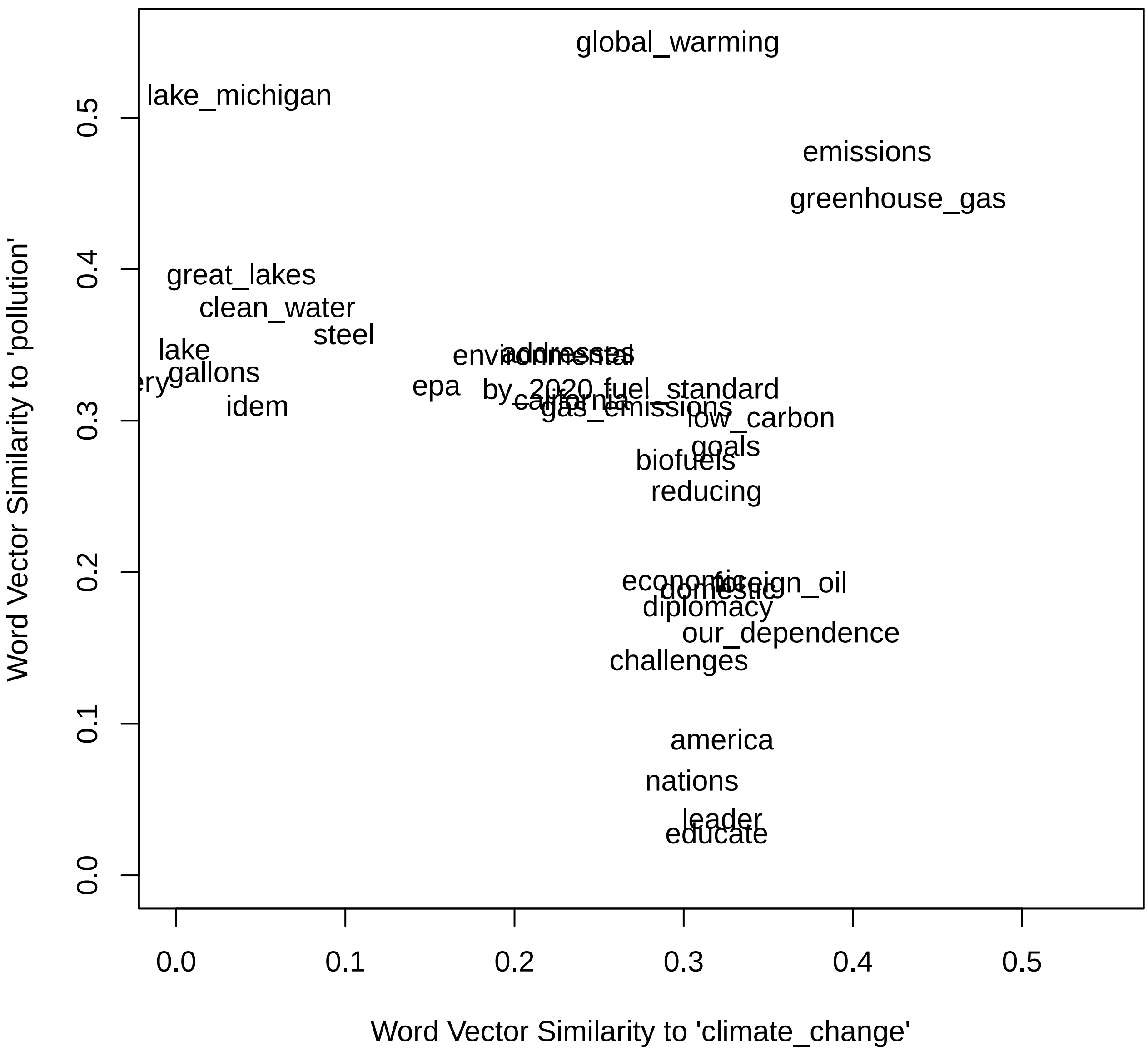
global_warming
lake_michigan
emissions
greenhouse_gas
great_lakes
clean_water
steel
lake
gallons
idem
epa
environmental
addresses
by_2020
fuel_standard
california
gas_emissions
low_carbon
goals
biofuels
reducing
economic
foreign_oil
domestic
diplomacy
our_dependence
challenges
america
nations
leader
educate
0.0
0.1
0.2
0.3
0.4
0.5
Word Vector Similarity to 'climate_change'
Word Vector Similarity to 'pollution'

Figure 4:

the closest words to the vector that results from subtracting the word vector for “poor” from the word vector for “tax” is “cheat,” “foreign_subsidiary,” “reward_companies,” “an_unfair,” and “bad_actors.” It’s easy to see that the way that Obama talked about taxes and the poor is quite different from the way in which he talked about taxes for the wealthy.

### 3.7.2 Document Embeddings

While word embeddings retain more information about an individual word, they come at a cost – instead of representing documents, like the document-feature matrix we constructed at the beginning of this chapter did, we have now produced representations of words across documents, or a matrix that retains a vector for each unique word within the vocabulary. Because most of the tasks we are interested in in this book are at the document level, we need a way to aggregate the word vector into documents – i.e. to use the information provided by the word embeddings to describe the documents.

There are several ways to aggregate word vectors at the document level. One is to simply take a weighted average of the word vectors that are included within the document, or by some other aggregation process. When this is applied, researchers might use word vectors pre-trained on a very large corpus like Wikipedia to capture word usage in the language as a whole, then aggregate these vectors to describe the documents in their corpus. Or, they can train word vectors within their specific corpus and then aggregate to the document level.

Alternatively, document vectors can be learned explicitly from the data, as in doc2vec (Le and Mikolov, 2014). doc2vec employs a similar model as the skip-gram algorithm, but in addition to the word vectors includes a document-level vector in the prediction model for $y$. Similarly, the aggregation of words can be an explicit part of a method for classification, as often used in neural networks (described in Chapter 5). Either way, once documents are represented by vectors, we can then perform computations on the document vectors, including computing the similarities between pairs of documents, as discussed before.

## 3.8 What is lost in quantitative analysis? How could this ever work?

When we turn text into numbers, we lose a lot of information. On the one hand, discarding information helps us – it allows us to make the problem of analyzing text as scale tractable. On the other hand, we lose a lot of nuance when we discard information. When we simply count the number of times each document uses "love," we aren't sure which of those documents use it genuinely, "I love ice cream," and which use it facetiously, "We all love homework." Indeed, these two loves are counted similarly to each other. We also discard the word order – typically more emphasis is placed on words in the beginning or end of a document, in the introduction or conclusion, for example. Or maybe one of the words was in a heading in 24-point bold font, while another was relegated to a footnote. Regardless of their placement, these words are typically still treated equally.

Much about the context of the document we also lose. The words within the text are only that – words – and they may mean something very different in different time periods, social, and political contexts. Infamously, "dog whistles" are identical words or phrases that have two meanings based entirely on the audience, and we may not know the types of people who read the text or what the author intended for them to mean. Under political, social, or economic pressure, authors may also be indirect, beat around the bush, or use wordplay or homophones to get their point across in almost "code" and these elements of the text might be lost on the analyst.

So how could this ever work? One answer is that it might not. Throughout this book, we put heavy emphasis on validation – reading the text to make sure that whatever method of text compression is used to simplify it maps correctly back to the original texts at hand, re-conducting the analyses in different ways and on different datasets to make sure that our results are general enough to be useful, and exploring all possible observable implications of the theory or argument. Only after extensive validation can we be sure that we have done any analysis as scientifically as possible.

While the steps we've outlined here lose a lot of information within text, we cannot lose sight of the fact that discarding complexity affords enormous advantages in the analysis of text. Simplifying the text allows us to organize, compare, and paint a broad picture of texts – which computers are good at – along dimensions we view as important and reserve more time and energy for what we as humans are good at – selecting a few texts to read and understand thoroughly. The point of much of text analysis methods is to characterize the haystack, to make a broad point about (sometimes) very large corpuses. If one straw of hay doesn't fit into the broad picture, that usually won't worry us, as long as we capture the general points in the text more broadly. If we make smart choices in sampling and representing texts quantitatively, we will have more confidence that we have painted the correct bird's eye view of our corpus.

To illustrate how the simplification and quantification of text can succeed at characterizing the haystack where humans would fall short, we report the results of an experiment first alluded to in King, Schlozman and Nie (2009) and first reported in Grimmer and King (2010).[1] The goal of the experiment was to organize essays in a book about the "future of political science" (King, Schlozman and Nie, 2009). To demonstrate that substantial information remains for this purpose of characterizing the haystack even after preprocessing texts, we begin with 100 diverse essays, each less than about 1,000 words and written by a political scientist. During copyediting, the publisher asked the editors to add to the end of each essay a "see also" with a reference to the substantively closest essay among the remaining 99. The editors agreed to allow Grimmer and King (2010) to conduct the following experiment which provides an extremely useful way to evaluate the preprocessing procedures discussed in this

[1]The results are from an unpublished working paper. We thank Gary King for permission to include these results.

chapter and compare them to the human alternative.

Grimmer and King (2010) begin with a random selection of 25 of the essays. For each, they chose the closest essay among the remaining 99 in the entire book in four separate ways. First, they used the quantitative recipe described in this chapter, first preprocessing the texts, then finding the most similar essay to each of the 25 based on the cosine similarity metric, described earlier in this chapter. Second, a group of graduate students and a faculty member at one university chose, for each of 25 randomly selected essays, the closest other essay they could find among the remaining 99. Third, another set of graduate students and a faculty member at a different university grouped together the essays into clusters of their choosing, and then pairs were created by randomly selecting essays within their clusters. And finally, as a baseline, Grimmer and King (2010) matched each essay with a randomly selected essay. They then randomly permuted the order of the resulting $25 \times 4 = 100$ pairs of documents, blinded information about which method was used to create each pair, and sent them back to one graduate student from each group to evaluate. Each graduate student then scored each pair separately as (1) unrelated, (2) loosely related, or (3) closely related.[2]

Using the same evaluators likely biases the results against the automatically created pairs, but Grimmer and King (2010) found that the machine triumphs anyway. Table 4 presents the results, with values indicating the average on this scale from 1 to 3. The first row gives the machine-created pairs which are closest among all methods: it is consistently higher the two control groups created by humans, which in turn is higher than the randomly constructed pairs.

The human evaluators thus judge the automatically created pairs to be more closely related than the pairs they themselves created. This is a small experiment, but it confirms the validity of the standard approach to turning text documents into textual summaries,

[2]Extensive pretesting indicated that inter-coder reliability would suffer with more categories, but coders are able to understand and use effectively this simple coding scheme.

| Pairs from | Overall Mean | Evaluator 1 | Evaluator 2 |
|---|---|---|---|
| Machine | 2.24 | 2.08 | 2.40 |
| Hand-Coding | 2.06 | 1.88 | 2.24 |
| Hand-Coded Clusters | 1.58 | 1.48 | 1.68 |
| Random Selection | 1.38 | 1.16 | 1.60 |

Table 4: Evaluators' Rate Machine Choices Better Than Their Own. This table show how preprocessing steps accurately encode information about the content of documents and how automated approaches outperform hand coding efforts.

as well as the point made above that machines can easily outdistance humans in extracting information from large quantities of text. Even though much information is lost when we preprocess texts, this serves a good purpose – allowing us to characterize the main themes of the text in a way that allows humans to better leverage the text. With proper validation, this simplification puts us well on our way to employing text analysis techniques for the purpose of social science.

# 4 Conclusion

In this chapter, we described the process of getting started with using text quantitatively for social science research. We discussed how to identify and select texts, including a check list that researchers can use to remind them of common sources of bias that exist for text corpuses. Next, we discussed the process of turning text into numbers – tokenizing, eliminating complexity, and creating the document-term matrix. We then introduced a number of techniques for extracting more complexity out of text – from parts of speech tagging to dependency parsing. We discussed how to use documents as vectors and find similar documents, and we talked about word embeddings, or encoding more information about the words than simply whether they occur in the document. With these tools, you should be able to start working with texts, moving onto the next task: discovery.

# References


Abello, James, Peter Broadwell and Timothy R Tangherlini. 2012. "Computational folkloristics." *Communications of the ACM* 55(7):60–70.

Asur, Sitaram and Bernardo A Huberman. 2010. Predicting the future with social media. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on.* Vol. 1 IEEE pp. 492–499.

Barberá, Pablo and Gonzalo Rivero. 2014. "Understanding the political representativeness of Twitter users." *Social Science Computer Review* p. 0894439314558836.

Beieler, John. 2016. "Creating a Real-Time, Reproducible Event Dataset." *arXiv preprint arXiv:1612.00866* .

Bengio, Yoshua, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin and Jean-Luc Gauvain. 2006. *Neural Probabilistic Language Models.* Berlin, Heidelberg: Springer Berlin Heidelberg pp. 137–186.

Blaydes, Lisa, Justin Grimmer and Alison McQueen. 2018. "Mirrors for princes and sultans: advice on the art of governance in the medieval Christian and Islamic worlds." *Journal of Politics* (4). Forthcoming.

Boydston, Michelle D. 1991. "Press censorship and access restrictions during the Persian Gulf War: A First Amendment analysis." *Loy. LAL Rev.* 25:1073.

Danzger, M Herbert. 1975. "Validating conflict data." *American Sociological Review* pp. 570–584.

Denny, Matthew James and Arthur Spirling. 2018. "Text Preprocessing for Unsupervised Learning: Why It Matters, When It Matters, and What To Do About It." *Political Analysis* .

Dodds, Peter Sheridan, Kameron Decker Harris, Isabel M Kloumann, Catherine A Bliss and Christopher M Danforth. 2011. "Temporal patterns of happiness and information in a global social network: Hedonometrics and Twitter." *PloS one* 6(12):e26752.

D'Orazio, Vito, Steven T Landis, Glenn Palmer and Philip Schrodt. 2014. "Separating the wheat from the chaff: Applications of automated document classification using support vector machines." *Political analysis* 22(2):224–242.

Franco, Annie, Justin Grimmer and Monica Lee. 2016. "Changing the Subject to Build an Audience: How Elected Officials Affect Constituent Communication.".

Gill, Michael and Arthur Spirling. 2015. "Estimating the Severity of the WikiLeaks US Diplomatic Cables Disclosure." *Political Analysis* 23(2):299–305.

Golder, Scott A and Michael W Macy. 2011. "Diurnal and seasonal mood vary with work, sleep, and daylength across diverse cultures." *Science* 333(6051):1878–1881.

Graesser, Arthur C., Danielle S. McNamara, Max M. Louwerse and Zhiqiang Cai. 2004. "Coh-Metrix: Analysis of text on cohesion and language." *Behavior Research Methods, Instruments, & Computers* 36(2):193–202.
**URL:** *https://doi.org/10.3758/BF03195564*

Grimmer, Justin. 2013. *Representational Style in Congress: What Legislators Say and Why It Matters.* Cambridge University Press.

Grimmer, Justin and Brandon M Stewart. 2013. "Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts." *Political analysis* 21(3):267–297.

Grimmer, Justin and Gary King. 2010. "A General Purpose Computer-Assisted Document Clustering Methodology.". http://gking.harvard.edu/files/abs/discovm-abs.shtml.

Gup, Ted. 2008. *Nation of Secrets: The Threat to Democracy and the American Way of Life.* Anchor.

Handler, Abram, Matthew J Denny, Hanna Wallach and Brendan O'Connor. 2016. "Bag of what? simple noun phrase extraction for text analysis." *NLP+ CSS* 114.

House, Freedom. 2015. *Freedom on the Net 2015:Privatising Censorship, Eroding Privacy.*

Jaros, Kyle and Jennifer Pan. Forthcoming. "China's Newsmakers: Official Media Coverage and Political Shifts in the Xi Jinping Era." *China Quarterly* .

Jursfsky, Dan and James Martin. 2009. *Speech and natural language processing: An introduction to natural language processing, computational linguistics, and speech recognition.* Upper Saddle River: Prentice Hall.

King, Gary, Jennifer Pan and Margaret E. Roberts. 2013. "How Censorship in China Allows Government Criticism but Silences Collective Expression." *American Political Science Review* 107:1–18. http://j.mp/LdVXqN.

King, Gary, Kay Schlozman and Norman Nie, eds. 2009. *The Future of Political Science: 100 Perspectives.* New York: Routledge Press.

King, Gary, Patrick Lam and Margaret E Roberts. 2014. "Computer-Assisted Keyword and Document Set Discovery from Unstructured Text.".

Lampos, Vasileios and Nello Cristianini. 2010. Tracking the flu pandemic by monitoring the social web. In *2010 2nd International Workshop on Cognitive Information Processing.* IEEE pp. 411–416.

Le, Quoc and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning*, ed. Eric P. Xing and Tony Jebara. Vol. 32 of *Proceedings of Machine Learning Research* Bejing, China: PMLR pp. 1188–1196.
**URL:** *http://proceedings.mlr.press/v32/le14.html*

Lodhi, Huma, Craig Saunders, John Shawe-Taylor, Nello Cristianini and Chris Watkins. 2002. "Text classification using string kernels." *Journal of Machine Learning Research* 2(Feb):419–444.

MacKinnon, Rebecca. 2008. "Flatter world and thicker walls? Blogs, censorship and civic discourse in China." *Public Choice* 134(1-2):31–46.

Malesky, Edmund, Paul Schuler and Anh Tran. 2012. "The adverse effects of sunshine: a field experiment on legislative transparency in an authoritarian assembly." *American Political Science Review* 106(04):762–786.

Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*. pp. 55–60.
**URL:** *http://www.aclweb.org/anthology/P/P14/P14-5010*

Manning, Christopher D., Prabhakar Raghavan and Hinrich Schütze. 2008. *Introduction to Information Retrieval.* NY: Cambridge University Press.

Marcus, Mitchell P., Mary Ann Marcinkiewicz and Beatrice Santorini. 1993. "Building a Large Annotated Corpus of English: The Penn Treebank." *Comput. Linguist.* 19(2):313–330.
**URL:** *http://dl.acm.org/citation.cfm?id=972470.972475*

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, ed. C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani and K. Q. Weinberger. Curran Associates, Inc. pp. 3111–3119.
**URL:** *http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf*

Miller, George A., Richard Beckwith, Christiane Fellbaum, Derek Gross and Katherine J. Miller. 1990. "Introduction to WordNet: An On-line Lexical Database*." *International Journal of Lexicography* 3(4):235–244.
**URL:** *+ http://dx.doi.org/10.1093/ijl/3.4.235*

Mosteller, Frederick and David L Wallace. 1963. "Inference in an authorship problem: A comparative study of discrimination methods applied to the authorship of the disputed Federalist Papers." *Journal of the American Statistical Association* 58(302):275–309.

Nivre, Joakim, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira et al. 2016. Universal Dependencies v1: A Multilingual Treebank Collection. In *Tenth International Conference on Language Resources and Evaluation (LREC 2016)*.

Norris, Pippa. 2001. *Digital divide: Civic engagement, information poverty, and the Internet worldwide.* Cambridge University Press.

O'Connor, Brendan, Brandon M. Stewart and Noah A. Smith. 2013. Learning to Extract International Relations from Political Context. In *Proceedings of the 51st Annual Meeting of the*

*Association for Computational Linguistics (Volume 1: Long Papers)*. Sofia, Bulgaria: Association for Computational Linguistics pp. 1094–1104.
**URL:** *http://www.aclweb.org/anthology/P13-1108*

O'Connor, Brendan, Ramnath Balasubramanyan, Bryan R Routledge and Noah A Smith. 2010. "From tweets to polls: Linking text sentiment to public opinion time series." *ICWSM* 11(122-129):1–2.

Porter, Martin F. 1980. "An algorithm for suffix stripping." *Program* 14(3):130–137.

Schofield, Alexandra and David Mimno. 2016. "Comparing Apples to Apple: The Effects of Stemmers on Topic Models." *Transactions of the Association for Computational Linguistics* 4:287–300.

Schofield, Alexandra, Måns Magnusson and David Mimno. 2017. "Pulling Out the Stops: Rethinking Stopword Removal for Topic Models." *EACL 2017* p. 432.

Schonhardt-Bailey, Cheryl. 2017. *Accountability, Oversight and Deliberation of Economic Policy in UK Parliamentary Committees.*

Snyder, David and William R Kelly. 1977. "Conflict intensity, media sensitivity and the validity of newspaper data." *American Sociological Review* pp. 105–123.

Spirling, Arthur. 2012. "US treaty making with American Indians: Institutional change and relative power, 1784–1911." *American Journal of Political Science* 56(1):84–97.

Strapparava, Carlo and Rada Mihalcea. 2008. Learning to Identify Emotions in Text. In *Proceedings of the 2008 ACM Symposium on Applied Computing.* SAC '08 New York, NY, USA: ACM pp. 1556–1560.
**URL:** *http://doi.acm.org/10.1145/1363686.1364052*