VISUALIZING AMERICAN POLITICAL SPEECH IN THE 21ST CENTURY

By

CASSIDY TAYLOR MARTIN

A Thesis Submitted to The Honors College
In Partial Fulfillment of the Bachelors degree
With Honors in

Computer Science

THE UNIVERSITY OF ARIZONA

MAY 2018

Approved by:

Dr. Kate Isaacs
Department of Computer Science
# Table of Contents

Abstract.......................................................................................................................... 1  
Introduction................................................................................................................................ 1  
Related Works....................................................................................................................... 2  
  Coxcombs ............................................................................................................................ 2  
  Better Life Index Flower ................................................................................................. 3  
  EvoRiver ............................................................................................................................ 4  

Data ...................................................................................................................................... 5  

Data Processing.................................................................................................................... 6  
  Pulling Transcripts from American Rhetoric Speech Bank ............................................ 7  
  Topic Modeling ..................................................................................................................... 8  
  Calculating Word Frequencies ............................................................................................. 10  

Visualizing the Data.............................................................................................................. 11  
  Case Studies ....................................................................................................................... 13  

Discussion................................................................................................................................ 15  

Conclusion............................................................................................................................ 16  

Works Cited.......................................................................................................................... 17  

Appendix A.......................................................................................................................... 17  

Appendix B.......................................................................................................................... 20
Abstract

In order to see how political speech has changed over time in the United States, I created a data visualization project that takes speeches made by U.S. politicians and shows the usage of popular topics and the words in those topics. My project focuses solely on the 21st century and uses a flower design to represent each topic. Using transcripts of political speeches, along with natural language processing and topic modeling tools, I was able to create a visual representation of what politicians talk about in the U.S. and how those topics have changed.

Introduction

In our current political environment, an emphasis has been placed on political speech. I wanted to see how political speech has changed over time in the United States, specifically what our politicians talk about. Current tools that show political speech are either not designed for use by people who are not researchers in their field or they don’t look at the progression of speech over time. I wanted to create a tool that was simple and easy to understand that would track the changes in political speech over several years. To do this, I collected transcripts of speeches made by United States politicians. I used topic modeling to find topics and words in those topics to analyze and then I calculated the frequency that those words were used in speeches made by U.S. politicians in the 21st century. Then I took that data and made it into an interactive visual so that the data would be easy to explore and I could see how political speech in America has changed.

The first section of this paper discusses related tools and visualization that already exist and why I made different choices when creating my tool. The second section discusses the data and how it was collected. The third section discusses how I decided which topics and words to
display in my tool and how I collected data on those words from the speeches I collected. Finally, the fourth section focuses on the visualization itself and what it shows.

**Related Works**

There are several visualizations that already use the flower shape similar to the one I ended up using in my tool. My visualization differs from its predecessors both in how it uses only length to represent differences in data and the way it allows data to be compared over time. There are also tools that have displayed words grouped by topic. I have discussed some of these tools in more detail below.

**Coxcombs**

The coxcomb visualization was created by Florence Nightingale during the Crimean war to highlight the proportion of soldiers dying in hospitals due to disease instead of injury (ims25, Nightingale's 'Coxcombs'). Coxcombs are a circular graph which compares different categories and their values by representing each category with part of a circle (ims25, Mathematics of the Coxcombs). All categories span the same angle in the circle, but the value for each category is reflected in the area of its portion of the circle (ims25, Mathematics of the Coxcombs). The radius is adjusted in each portion so that the area of each portion reflects the value associated with the portion (ims25, Mathematics of the Coxcombs). An example of a coxcomb can be seen in Figure 1. Coxcombs have a similar shape to the visualization I created for this project in that each topic in my visualization has an origin point and the words in that topic all come out of that origin point at evenly distributed angles. However, my visualization uses the radius of each word’s line to compare values and not area, like a coxcomb.
Figure 1: The coxcomb design uses the area of the sections to indicate differences in data. Although my design has a similar radial shape, it focuses on the length of the sections to compare data values.

Better Life Index Flower

The Better Life Index Flower visualization was created by the Organisation for Economic Co-operation and Development. The purpose of the visualization is to rank countries on 11 topics and how a specific user values those topics. Each user ranks the topics Housing, Income, Jobs, Community, Education, Environment, Civic Engagement, Health, Life Satisfaction, Safety, and Work-Life Balance by order of personal importance. Then, a visualization is created with one flower per country where each petal is one of the 11 topics, the width of the petal signifies how important the topic is to the user, and the length of the petal signifies how well the country scored in that topic (Organisation for Economic Co-operation). A country with long, thick petals and short skinny petals would be the best match for the user. An example of the Better Life
Index Flower can be seen in Figure 2. This visualization has a similar shape to mine, although I do not use petal width to signify anything about my data, and it is designed to be used by the general public, not specifically by experienced researchers. Also, this visualization cannot be compared over time, while my visualization is designed to do so.

![Image of Index Flower](image)

*Figure 2: The Better Life Index Flower uses both petal length and petal width to display a countries score for a topic and the user’s preference for that topic respectively. My visualization has a similar flower shape, but focuses solely on the length of the petals (Organisation for Economic Co-operation).*

**EvoRiver**

EvoRiver (Figure 3) was designed to show patterns in social media data over time, specifically which topics cooperate and compete with each other (Sun, Wu and Liu). This visualization is similar to mine because it allows data to be seen over time so that patterns can be detected. It also takes in data and groups it by topic. However, it is different from mine in the way the data is displayed. EvoRiver maps the relationships between topics as well as displaying the topics. My visualization allows patterns to be seen between topics but does not define the types of relationships between the topics or distinguish between types of topic relationships.
Also, EvoRiver is not intended to be used by the general public because it requires users to understand the ideas of cooperation and competition of topics, as well as determine the causes of shifts in relationships on their own.

![Diagram of EvoRiver](image)

*Figure 3: EvoRiver takes data from social media over time and shows which topics cooperate and compete with one another. This visualization is similar to mine in that it groups speech by topic and shows both the topics and the words in those topics, but my design does not focus on classifying the relationships between the topics and displays the temporal aspects of the data using animation rather than timelines (Sun, Wu and Liu).*

**Data**

To study the speech of United States politicians, I looked for transcripts of their speeches so that I could analyze the words they used. The American Rhetoric Online Speech Bank proved to be a useful resource. It stores thousands of transcripts of speeches made by prominent American public figures throughout history. I found this speech bank to be the best fit for my project because it contained a large number of transcripts of speeches from United States politicians and all of the transcripts were in a similar format so they would be easy to process. Each transcript had the name of the author, the name of the speech, the location the speech was given at, and the date it was given, along with the body of the speech. I only allowed speeches from United States Presidents, Congressmen, Cabinet Members, and Supreme Court Justices. In
total, my data set contained the transcripts of 602 speeches given between the years 2000 and 2017. I focused on speeches given in the 21st century only because they were the years with the most speeches. After I collected the transcripts initially, I graphed the number of speeches by year to see how the speeches were distributed over time as seen in Figure 4. Before the year 2000, the number of speeches per year was not high and I did not think it would be appropriate or accurate to represent a whole year of political speech with so few speeches.

![Graph of Number of Speeches in Dataset Per Year](image.png)

*Figure 4: The number of speech transcripts collected for each year.*

**Data Processing**

In order to make the transcripts easier to work with, I scraped them off of the American Rhetoric website and saved them as text files.
Pulling Transcripts from American Rhetoric Speech Bank

I pulled transcripts from 6 pages on the American Rhetoric Website: the Barack Obama Speech Bank, the George W Bush Speech Bank, and the Complete Speech Bank which was broken up into 4 pages based on the first letter of the first word in the title of each speech’s page (usually the first letter of the author’s first name). The reason I pulled from the Barack Obama and George W Bush speech banks as well as the complete speech bank is that the complete speech bank only had a small selection of each author’s speeches while the individualized speech banks had the whole collection of transcripts. Each speech on the American Rhetoric website has its own webpage. I wrote a web scraper that found all of the speeches linked to on the six 6 pages I pulled from and scraped information from each of those pages. My scraper ignored speeches that did not reside on a page on the American Rhetoric website because I could not think of a reasonable way to scrape the information I wanted from external webpages which all had different formatting. I also only scraped speeches from specific speakers. The American Rhetoric Speech Bank does not have a specific speech bank for only politicians, so I had to figure out which speakers were politicians and pull speeches from only those speakers. I also paired it down from there because there were a lot of different formatting types for speaker names on the pages so I choose the speakers with multiple speeches and used only those speakers. This was a way to make dealing with the different formats manageable for my timeline. For each speech, I collected the name of the speaker, the name of the speech, the date the speech was delivered, the location the speech was given at, and the transcript. I choose those fields because they were available for all of the speeches, with the exception of the location field. Because of the formatting of the pages the speeches were on, I was able to pull in a location for a speech at the same time I pulled in the date, so I decided to keep the location data in case I wanted to use it
later. If a location was not available for a given speech, I made note of that. The scraper saved
the data for each speech in a separate text file so that I could process that data later.

All of the code for the scraper was written in Python and I used the Beautiful Soup
library to navigate the html code from the transcript’s webpage. I decided to use the Beautiful
Soup library because it provided a small, but powerful group of functions that allowed me to
navigate the html code on a webpage and easily create lists of text based on html tag type. This
worked to my advantage because I was not familiar with python or web scraping before this
project, so the manageable size and shallow learning curve allowed me to make progress on my
project without having to spend too much time researching syntax and reading through a large
amount of documentation. For more specific information on how Beautiful Soup was used in this
project, please see Appendix A.

Topic Modeling

Since I wanted to examine how the frequency of word usage changed over time for
certain words, I used a topic modeling program to identity words that were related to one another
and used often in the dataset. I used the Gensim and NLTK libraries for Python to write a
program that performed a topic modeling analysis on the transcripts. Gensim is an open source
topic modeling toolkit that, among other things, allows users to perform topic modelling with an
LDA model on a large quantity of documents in a short period of time. The LDA (Latent
Dirichlet Allocation) model works by assuming a document is made up of a few topics and tries
to figure out which topics represent the collection of documents by repeatedly moving words into
different topics based on the other words in that topic and how all of the documents represent
those topics based on the current words in the topic (Chen). The Gensim LDA model lets the
user choose the number of topics and the number of words in each topic that appear in the
Having fewer words in a topic will make that topic more general. The same goes for the number of topics. If there are fewer possible topics, the topics will be broader to fit more of the documents’ contents accurately. I used 2 combinations of topic number and word number to get the output that I used in the final product of this project. I used 5 topics with 50 words each and 7 topics with 100 words each. I started with the first combination, but found that the topics were too broad, so I tried the second combination and was more satisfied with the results. I did not use all of the words and topics that came out of the topic modelling program. Instead, I looked at each topic that came out and looked for a common theme among the words in the topic. If I could find a recognizable theme for the words, I made that the topic name since the LDA model does not name the topic itself. Then, I removed words that were common in all of the topics or I kept it in only one topic where I thought it was most relevant and I removed words that were common in the English language and not interesting to observe. For example, words like “their” were removed because they are common and uninteresting and words like “man” were removed because they showed up in almost all of the topics. In the end I ended up with 4 topics: War/Foreign Policy, Law, Economy/Jobs, and School Shootings/Violence. The list of words for each is as follows:

- **War/Foreign Policy:** Iraq, terrorist, national security, war, military, force, threat, attack, Al Qaeda, Afghanistan, nuclear, Iran, ISIL
- **Law:** constitution, court, government, right, law, judge, principle, free, case, federal, fact, congress, decision, compromise
- **Economy/Jobs:** job, health care, tax, business, economy, work, reform, company, insurance, cost, energy, pay
• School Shootings/Violence: time, young, family, child, help, community, gun, school, life, thought, prayer, faith, violence

For more information on how the Gensim and NLTK libraries were used in this project, see Appendix B.

Calculating Word Frequencies

Now that I had my topics and a list of words for each topic, I calculated how frequently each word was used for each year and then multiplied that by how much the speeches from that year represented the total speech collection. To do this, I counted the number of times each word was used for each year and then divided that number by the total number of words for each year. This number would be the percent usage of the word for the year or what percentage of all the words in the year were the specific word. Then I calculated how many words were in all of the speeches in all of the years and divided the number of speeches in a specific year by the number of speeches in all of the years to figure out how much one year represented the total collection of speeches. I multiplied the percent usage of a word in a year by that year’s percentage out of all of the years to get my data for the visualization.

\[
wordValue = \left( \frac{\text{# of word uses in year}}{\text{# of words in year}} \right) \times \left( \frac{\text{# of words in year}}{\text{# of words in all years}} \right) \times \text{constant}
\]

These values were calculated for each word in each year and stored in a csv file with one row per word per topic. Each row listed the topic, the word, and the word’s frequency for each year. I decided to use this method because I wanted all of the topics in one year to be comparable, which is why I looked at a word’s percent usage in a year, but I also wanted a flower in multiple years to comparable, so I needed to weight years with more speeches heavier so that the visualization wouldn’t be misleading. For example, if a year with 100 words used the word
“flower”, it would be 1% of that year’s words and if a year with 1000 words used “flower” 10 times, it would also be 1% of that year. Since “flower” is used more in the second year, it should appear larger in the visualization, so I weighted the years. Since the second year makes up more of the total words in all of the years, it is weighted heavier than the first year so that when the years are compared, the difference between word frequency can be seen.

**Visualizing the Data**

The visualization is based off of the idea of flowers. Each topic is a separate flower and each word is a petal. The petals have different lengths based on the weighted percent usage calculated earlier for the word for the year. The percent usage number is scaled by a constant in order to make the number large enough to be easily viewed. The scaled numbers are proportional to the original numbers, so it does not affect the meaning of the data when it is viewed. For each petal, the word it represents is shown at the end of the petal. The length of the petal represents how frequently the word was used as shown in Figure 5. I chose the flower design because I wanted the petals in each flower to be comparable, but I also wanted the flowers themselves to be comparable in each year and between different years. The flower allows each topic to be viewed on its own and also in comparison to other topics in the same or different years. Since each topic has its own flower, the size of the flower represents the frequency that topic was discussed in a given year. Thus, the user can see how much a topic was discussed relative to another topic in the speeches for a certain year. The user can also compare words from different topics, although subtle differences in word usage would be hard to detect since the petals might not be visually close to one another on the screen. To switch from year to year, there are radio buttons on the top of the page that transition the current year the user is viewing to any other year.
the user wishes to view. The flowers are animated so that the transition between the years is smooth. An example of the visualization can be seen in Figure 6.

Figure 5: The law flower from the year 2012. The length of each petal represents how frequently the word was used. The word “right” was used more frequently than the word “government”.

Figure 6: The visualization shown for the year 2012. The sizes of the flowers can be compared. For example, economy/jobs was a more discussed topic than war/foreign policy in this year. This is shown because economy/jobs has more words with longer petals.
Case Studies

There are clear connections between important events in U.S. history and the values displayed for the topics. For the War/Foreign Policy topic, there is a significant increase in word usage for most of the words in the topic in 2001, especially in words like terrorist and war as shown in Figure 7 and Figure 8. Afghanistan also saw an increase. Iraq didn't see an increase until a year later and ISIL didn’t see an increase until 2014. These increases line up with the time of important events like the attacks on September 11th, 2001, the start of the Iraq War in 2003, and an increase in ISIL activity in 2014 (CNN Library). This also appears in other topics. For example, in the Economy/Jobs topic the words job and economy saw significant increases in 2008 when the Great Recession started as seen in Figure 9 and Figure 10. The word gun saw spikes in 2013 and 2016, which could line up with the Sandy Hook shooting which happened at the end of December in 2012 and the Pulse Nightclub Shooting which happened in 2016. For a lot of the big spikes shown in the data I’ve collected, it is easy to find a significant current event relating to that topic in that year. I think this reflects the fact that politicians are expected to make statements after big events to reassure the public and give them the sense that the government has a plan of action in place. Also, big events capture the public’s attention after they happen and bring more people into the conversation about a topic that they might not have joined before. It is also interesting to see that some words do not experience spikes during current events. For example, national security never experiences any major spikes even during wars or the 9/11 attacks. Its usage stays mostly stable. Even though the rest of its topic sees an increase, it does not. The words compromise and constitution also stay stable.
Figure 7: The war/foreign policy flower in the year 2000. This was not a heavily discussed topic in 2000 because the petals are short.

Figure 8: The war/foreign policy flower in the year 2001. Most of the words experience a spike in 2001, as shown by the significantly longer petals than those in 2000.
Discussion

I think this tool could be expanded to look at smaller units of time, such as months. This could better confirm if certain current events are responsible for spikes in word usage. To do this, there would need to be more speeches to examine or each month would not have enough data for the visualization to be an accurate representation of that month. Also, the current way this project
measures data is by taking the percent usage of a word in a year and multiplying it by the percent usage of a year in the whole data set. This favors years with more words over years with fewer words. Another way you could compare the data is to just look at the percent usage of a word in a year, although that would favor smaller years over larger years. It could be interesting to compare the two methods and see what differences arise. I think this visualization succeeds in its goal to be easy to understand and read. I wanted to create a tool that could be seen by anyone, regardless of their knowledge of the subject, and still be understood. I think the simplicity of my design works to that end. I think there are many ways this project could be expanded upon, but as it is now, it serves the purpose of the tool I set out to create.

**Conclusion**

I set out to work on this project because I wanted to see how political speech was changing in the United States. I think I achieved my goal to some extent. In the future, I think this project could benefit from more data. I had to overlook many speeches that had formatting issues because of time constraints. I also had to focus solely on the 21st century because I did not have enough data from before that time period. I would like to find more speeches from before 2000 and see what topics might be notable and how speech has changed over a larger span of time. I would also like to work more on the flower visualization and make the petals more interesting looking and work on the positioning of the petal labels so that they could be larger without overlapping. Through this project, I’ve seen how current events shine a spotlight on different topics. A major event brings up topics that might not be discussed very often and makes them open for discussion by more people. It was interesting to see how much current events influence what our politicians say and the national conversation as a whole. Overall, this project helped me learn about natural language processing, topic modeling, visualization, and political
speech as a whole. I believe that what politicians talk about shows what is important to them and what they believe is important to their constituents. Understanding how what they talk about has changed can help us understand not only what our government is focused on, but also what the American people are focused on.

Note: All data and code used in this project can be found at

https://bitbucket.org/ctmartin/political-rhetoric-vis

Works Cited


Appendix A

I pulled speeches from 6 pages, each of which was formatted as a table of links to pages with one transcript per page. I wrote a web scraper that found all of the links in the table and
stored them in a list and removed links that went to external sources as shown in Figure 11. The scraper also removed any duplicate links so that no speech could be scraped twice.

```python
# make a list of all the webpages with speech transcripts
urlList=["a-f.htm", "g-l.htm", "m-r.htm", "s-z.htm"]
list2=[]
for u in urlList:
    list=[]
    url="https://www.americannrhetoric.com/speechbank"+u
    page=requests.get(url, headers={'User-Agent':'Mozilla/5.0'})
    soup = BeautifulSoup(page.content, 'html.parser')
    # make a list of all a tag lines
    list=soup.findAll('a')

    # parse out url of transcript in each a tag line and save url in list 2
    for link in list:
        # parse out url
        if link.has_attr("href")== True:
            link=link.get("href")
            # find only links on American Rhetoric website
            if link.endswith(".htm")==True and link.startswith("speeches")==True:
                list2.append(link)
```

*Figure 11*

For each speech, I first scraped the author’s name. I did this by looking for b tags in the page’s html code. Unfortunately, there are small differences in the html code from speech to speech so when scraping the names of the authors, the name may or may not be split into multiple b tags. For example, in some cases the first letter of the speaker’s first and last names were split into separate b tags. This can be remedied by pulling the text from all of the b tags and removing newline and carriage return characters. As you can see in Figure 12, I set a flag for speaker names that have all of the name in one b tag and if it wasn’t all in one b tag, the flag was set so that it would pull in the name from four b tags and put them together to account for the speaker’s name being in the formatting mentioned in the example above.
Some of the speech authors went by several names, so I picked one of their names and replaced all of the other versions of their name with the one I had selected so that all of their speeches would be under the same name. I also had to filter out speeches who were spoken by people who were not politicians. To do this, I pulled a list of all of the authors and picked out which ones were politicians. I then checked to make sure the author of the speech I was scraping was on my list of politicians before I continued scraping the transcript. The next thing I scraped was the title of the speech. To do this, I pulled the text from the first i tag in the page’s html code. To collect the date and location, I iterated through all of the font tags on the page and looked for certain keywords that always precede the date on these pages. For example, the keyword for most of the speeches was “Delivered”. After I found a keyword, I would pull the text after the keyword as the date and location. The date and location are separated by a comma, so I would look for the comma and split the string around it. If there was no comma, I used the whole string as the date and made the location “no location listed”. There were some instances that the date/location line did not split properly and in those cases, I went through and manually
fixed the date and location in the text files for those speeches. There were not many speeches like this, so it did not prove to be a big inconvenience. The author’s name, the speech name, the date, and the location were printed on separate lines so that they made up the first four lines of the text file for the speech. Finally, I scraped the body of the speech by creating a start flag variable and an end flag variable. I saved all of the text in the tag when the start variable was true and the end variable was false. To set the start flag, I looked for the first p tag that did not have one of the keywords from earlier in it. The end flag was set when I found the first hr tag as seen in Figure 13. I wrote the text of the transcript to the text file underneath the line with the location.

```python
startFlag=False
dendFlag=False

for tag in soup.find_all():
    if tag.name=='hr':
        endFlag=True
    if startFlag==True and endFlag==False and tag.name=='font' and tag.has_attr('size') and tag['size']=="2":
        text=text+tag.get_text()
    for keyword in dateKeywordList:
        if (tag.getText().find(keyword) != -1) and tag.name=='p':
            startFlag=True
```

**Figure 13**

**Appendix B**

The first step in implementing the Gensim LDA model was to put the bodies of all of the transcripts into a document vector, where each cell of the vector had the transcript of the body of one speech. Next, I removed all of the stop words and punctuation from the transcripts in the document vector by using the NLTK library’s list of English stop words and punctuation. I also used the NTLK library to lemmatize all of the words in the document vector so that each word was changed into its base word without inflectional endings as shown in Figure 14. This made it so that words that were all the same base word would be recognized as the same word. This is important because if each word in each tense was counted as a different word, it would make it
hard to tell which words were actually used more. After all of the words in the document vector were lemmatized, I created a dictionary of the words, so that each unique word had an index.

That dictionary was then used to create a Document Term Matrix, which was fed into the Gensim LDA model as shown in Figure 15. This output the topic and word list, which I went through by hand and identified the words I wanted to use in my project.

```python
# Removing English stop words and punctuation. Lemmatizing the words.
stop = nltk.corpus.stopwords.words('english')
exclude = set(string.punctuation)
lemma = WordNetLemmatizer()
def clean(doc):
    stop_free = " ".join([i for i in doc.lower().split() if i not in stop])
punc_free = " ".join(ch for ch in stop_free if ch not in exclude)
normalized = " ".join(lemma.lemmatize(word) for word in punc_free.split())
    return normalized
doc_clean = [clean(doc).split() for doc in docMatrix]

Figure 14

# Creating the term dictionary of our corpus, where every unique term is assigned an index.
dictionary = corpora.Dictionary(doc_clean)

# Converting list of documents (corpus) into Document Term Matrix using dictionary prepared above.
doc_term_matrix = [dictionary.doc2bow(doc) for doc in doc_clean]

# Creating the object for LDA model using gensim library
lda = gensim.models.ldamodel.LdaModel

# Running and Trainign LDA model on the document term matrix.
ldaModel = Lda(doc_term_matrix, num_topics=7, id2word = dictionary, passes=100)

print(f"{ldaModel.print_topics(num_topics=7, num_words=100)}", file=textfile)

Figure 15
```