Political Speech Generation

Team Members:

Alex Lassalle, Timothy Contois, and Joshua Pikovsky

Abstract

Our work is focused on creating a statistical text generation model that uses a training set of speeches given by President Obama, and outputs a readable text based on an input of topic and Markov chain length. Our approach involves modifying a Markov text generation system to more accurately mimic the speech patterns and topical choices made by Obama in his speeches.This is accomplished through some light templating, and a collection of datasets tagged by topic, which are used to weigh certain words more heavily based on their association with the speech subject. The results vary based on the chain length chosen, but the output of the text generator is often somewhat readable and topically consistent, despite not always being completely coherent or grammatically sound.

Introduction

This project seeks to create a system that can replicate a realistic speech given by President Obama, and influence the word choice based on the selected topic. This is an interesting problem as it involves overcoming the initial difficulty of making a readable speech generator, and additionally finding a way to have the generated text bias its word choice to give the speech a sense of topic. The approach we took began with finding a large dataset of speeches by President Obama, which informs the basic structure of the text generated by the Markov model. We also compiled several smaller datasets with excerpts taken from speeches that were focused heavily on a specific topic, such as health care or foreign policy. When a topic is chosen, the model weighs the words in the associated topic dataset much more heavily and selects contextually appropriate words with a higher frequency. The research question that we are attempting to address is whether it is possible to use a statistical text generation approach to create a realistic sounding text that has an approximate resemblance to the speech patterns of President Obama,

and if this speech can be adjusted to discuss a particular topic. Our project has been very result-oriented, as each set of outputs informs us of the changes needed by the system. Depending on the chain length set, the outputs range from completely incoherent to direct quotes from the text. Our results have been interesting and unique, and they provide some insight into the power of the Markov model when coupled with modifications and expansions involving high-level templating and noun-phrase extractions.

Related Work

There are many other approaches to the problem of text generation. We chose an n-gram Markov model, but the other alternative we considered was a templating model. This works by creating pre-made templates of speeches with certain words omitted, and then filling in those spots based on the part of speech of the missing word. This works on the same principles as the game Madlibs, and it tends to produce much more coherent and predictable results. An example of a use for templating is found in the paper *Real vs. template-based natural language generation: a false opposition?* by Deemter, Krahmer and Theune. In their work they describe the canned-text template approach as being more practical and utilitarian, with its use generally fitting services like automated weather reporting or notifications of train departures **[1]**.

Another alternative technique would be to use Recurrent Neural Networks (RNNs), which are powerful models that are notoriously difficult to train. In Andrej Karpathy's blog post "The Unreasonable Effectiveness of Recurrent Neural Networks", recurrent neural networks are described as being powerful and robust, with many interesting applications. We eventually decided on the n-gram Markov approach because it was more conceptually interesting that templating, more realistic to complete in the timeframe of a semester, and had the potential for interesting and surprising outputs **[2]**.

A more general related work on natural language generation using Markov chains served as a starting point for our project. In his blog post "Generating pseudo random text with Markov chains using Python," Shabda Raaj explains how to use tuples of previous words as keys in a dictionary to store all words following the tuple for each key. Raaj also includes starter Python code to do this Markov chain text generation. We combined this Python Markov model (and our

own modifications, such as topic biasing) with a dataset of Obama speeches to make our text generator **[3]**.

Data

For our dataset, we have acquired a database of over 250 speeches by President Obama. The file is approximately 2.7 MB in size, and contains over 460,500 words and 25,000 sentences. This data comes from AmericanRhetoric.com, a site dedicated to compiling speeches by famous rhetoricians. The catalogue of Obama's speeches contains texts ranging from 2004 to 2015, and are typically between 2-3 pages in length. These are not tagged, but per the proposal feedback we have simplified our goal down to starting with just a simple generator from the position of one side. We excluded interviews and press conferences because they often involved Q&A, which would be tedious to sift through.

To bias the word choice made by the Markov generator when inputting a topic, we created subsets of the larger database that contain excerpts that are heavily related to each topic. These datasets are smaller, and are generally under 10,000 words. The content of these subject sets were handpicked based on their connection to the subject, and each set has an overall topic tag.

Our datasets can be found at https://github.com/timcontois/Datasets. The main dataset is "obama_speeches.txt" and the concentrated topic biasing files are "short_fp.txt" and "short_health_care.txt."

Method

1. Primary Approach

Our methodology began with finding a basic Markov model based text generation system **[3]**. This model is trained on our dataset of real Obama speeches. The model takes in a n-gram chain length size. As the model goes through the training data, it uses a tuple of the previous n-1 words as the key and adds the word to a list of words that follow this tuple. For example, assume we are using a 3-gram model on the sentence "I am John. I am James." For the tuple ("I", "am") both "John" and "James" will be stored at this key. When generating text, the model chooses the next word randomly from the list of words stored for the tuple of the previous n-1 generated words.

Once we'd generated a few sentences we began our modifications to the rudimentary text generation tool. The first change we made was to alter the method by which the model chose its first word. The initial system chose a word at random, but this was problematic as it would often choose a word from the middle of a sentence. To circumvent this, we added in rules so that would only choose words with capital letters that followed punctuation that would end a sentence. Once the model was picking starting words as the first word of the chain, the sentences started to make a little bit more sense. Another change we made was introducing an element of slight high level templating. Rather than using sentence templates, we applied the concept as a way to section off parts of the speech. For example, the beginning of the text generated is classified as the "greeting", and it is chosen from a set of premade greetings pulled from the text. Similarly, the ending of the generated text is the "closing", and those lines are also pulled from several closing lines Obama has used in his speeches. Once the model created text with a greeting, closing, and sentences that began with words that typically start a sentence, the output became significantly more readable.

Along with these rules, the readability was also greatly affected by the selected length of the Markov chain. We tested chains of lengths of two, three, and four to see which produced the output most suitable for this project. A chain length of two was clearly not enough and the output was basically total nonsense. When we tried a chain length of four, the output was almost perfectly coherent because it was essentially pulling quotes from the text. We decided to stick with a chain length of three because it produced output that still felt coherent while maintaining some sense of randomness and originality.

Once we had settled on a chain length and created some basic rules and templating, we focused our efforts on creating speeches that related to specific topics. To do this, we created subsets of the Obama database that were heavily concentrated excerpts relating to specific topics. During training, the words from these datasets would be added many times to the collection of possible words to use so that they would be weighed more heavily. When the model goes to generate a speech, it first selects a word from the concentrated topic file. As the generation goes on, the model is more likely to pick words related to the topic because the topic words have been added multiple times to the list it is randomly selecting from. This produces output that includes many

terms common to the selected topic. For example, training on actual Obama speeches stores both “Iran” and “credit card” in a list of possibilities following the tuple (“many”, “years,”). A model that weighs foreign policy will be much more likely to choose “Iran” as a noun, as opposed to “credit card” because “Iran” will have been added to the list multiple times during topic biasing.

2. Alternative Approaches

We explored several alternative methods of text generation to get a better sense of the limits of our model. One approach involved building up strings of part of speech tags, and then building a speech by generating words over each tag. The word choice would also be partially predicated on the prior two words, thus implementing a Markov chain approach to ensure that there is some sense of sequence in the word order. This approach is somewhat like a templating method, if each word was a blank space that was to be filled in. The results of these were not very impressive. Output seemed less coherent than the normal n-gram Markov model, and it seemed redundant to use parts of speech to pick words. This is because the n-gram Markov model already picked words that were contextually appropriate, since it chooses from a list of words that have followed the prior words in the chain. Topic biasing was not very successful either, mostly because the system was more focused on the part of speech tag as opposed to the topic dataset.

Another alternative method we pursued was using the modified Markov generator, but applying noun-phrase extraction tools to influence the word selection. This approach involved building a speech, and then using a noun-phrase identifier to highlight existing noun-phrases in the generated text. The noun-phrase tool would then be run through one of the topic datasets, and a random noun-phrase from the topical dataset would be selected to replace a noun-phrase in the generated text. The motivation was that it is noun phrases that can really influence the meaning of a speech. Here is an example of how the program ran:

<u>At first a speech is generated:</u>

“The implication is is that if it's a deal that can get done. But it is wonderful to be back with so many young men in prison, so many kids dropping out, so little hope. A hope gap. A hope gap

that still pervades too many communities all across the country. Rather than turn inward and wall off America from the rest of the hemisphere by over 50 percent."

The program then identifies potential noun phrases:

['young men', 'hope gap', 'hope gap', 'turn inward']

For each found noun phrase we replace them with a noun phrase randomly selected from our file of concentrated topic speeches, this is the result:

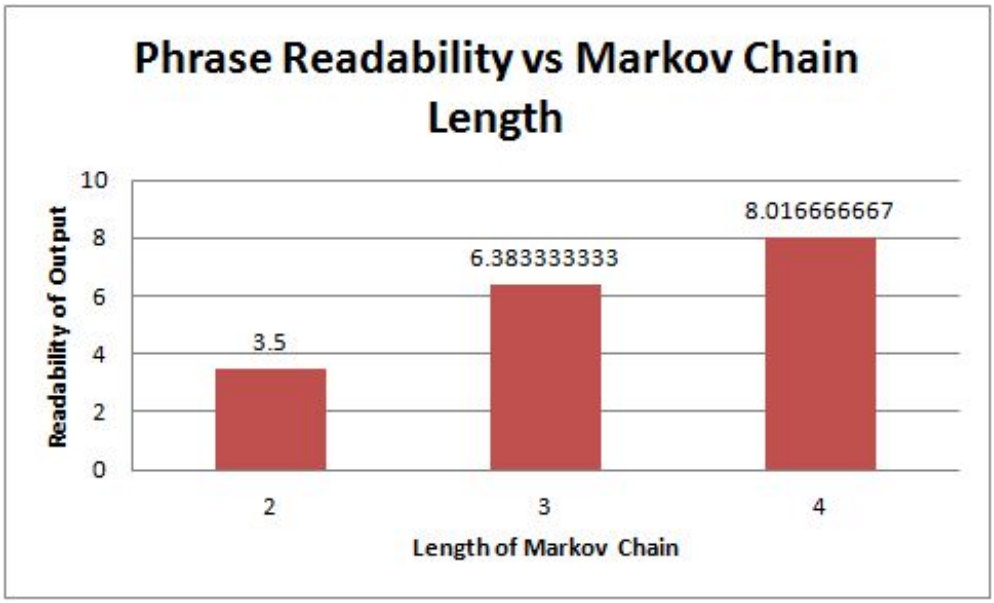
"The implication is is that if it's a deal that can get done. But it is wonderful to be back with so many CENTRIFUGE PRODUCTION FACILITIES in prison, so many kids dropping out, so little hope. A QAEDA. A PEOPLE ENDURES that still pervades too many communities all across the country. Rather than BRUTAL EFFECT and wall off America from the rest of the hemisphere by over 50 percent."

It is clear that noun phrase replacement is not a smooth substitution, and we hope in the future to develop laws when selecting a replacement noun phrase so that the readability is not negatively affected. The results of this approach showed some promise, as speeches began to include noun-phrases specific to certain topics. There was a high probability that noun phrases from other topics would be replaced with noun phrases from the desired topics, but we abandoned this approach because these inserted noun-phrases often didn't make sense in context.

Results

One of our first tasks involved determining how long of Markov chains to use in our model. To go about this, we generated sample output for chain lengths of 2, 3, and 4. We then individually graded the outputs on a 0-10 scale of readability, with 0 being gibberish and 10 being completely coherent. We calculated our average scores for the different n-gram lengths, which are included in the graph below.

Figure 1. Markov chain length average readability scores



To give a better sense of how the readability ratings correspond to the actual language, we have included a chart (Figure 2) that shows sample outputs, and how we graded the texts.

Figure 2. Analysis of Markov chain lengths and sample output

Chain Length	Example Output	Josh's Rating	Tim's Rating	Alex's Rating
2	I came to do. And to do more than the universe expanding training to missile defense in with other definition of speech can respond quickly how we could not another country.	4	3	4
3	We carry all that we would do. We can admit the intractability of depravation, and still strive for justice. We give thanks for that price. Every year, we freeze annual domestic spending, which represents a major commitment to comprehensive health care system.	6	8	7.5
4	I will not make that same mistake with health care. It was common ground, rooted in the cradle of civilization and a crucible of strife. For all the maps plastered across our TV screens today, and for their service and sacrifice.	9	6	8

Looking at the results, it would appear as though a length four Markov chain produces the most readable result. The example confirms that while those texts are the most readable, they are also the least original, as they tend to reconstruct quotes from the text. For example, one output with length four Markov chain produced the sentence "You know, it's been 12 weeks now

since my administration began. And I think it's important for members of Congress do, where -- we call it an "exchange," or you can face a growing challenge to its future." This sentence is pretty readable, but the first line "You know, it's been 12 weeks now since my administration began" is a direct quote from an Obama speech. There are many examples of sentences being drawn directly from the text with a length four Markov chain.

In addition to experimenting with different length Markov chains, we also experimented with alternative methods than our standard n-gram Markov chain with topic biasing. A major experiment we worked on was generating a sequence based on part of speech tags. First, we took part of speech sequences from our dataset using the nltk package and then generated a speech by emitting a word for each part of speech tag in the sequence. This had very poor results, so we switched to a part of speech templating intersected with Markov chaining method, i.e. at each step the list of words with the correct tag was intersected with the list of words following the previous tuple and the next word was chosen from the intersection. We also tried generating text and then finding and replacing noun phrases. The results for part of speech templating intersected with Markov chaining and noun phrase insertion are summarized below.

Figure 3. Alternative Methods and their Results

Alternative Method	Excerpt	Topic	Observations
part of speech templating with Markov chaining	"But they get back much to say it as, as I fight going to examine going for a of she to analyze up these health. Ive of your leaders, what are each of these least of all human, to be President to get into their school."	No Topic	● Less coherent than n-gram Markov ● Templating seems redundant
part of speech templating with Markov chaining	"Your kind enough just 's to professionalize new children in a vital in Medicare : to keep our word, to produce her stories, to reinforce racial in special hike that must ultimately do Our businesses is got."	Health Care	● Same as above ● Topic is not reinforced due to focus on part of speech tags
Noun Phrase Insertion	"And we've also seen a CALIFORNIA in Texas devastated by a TRUST FUND. and I want to discuss how we can reduce spending: by scouring the budget for the NEW WEBSITE for the arts. That's just not a HEALTH INSURANCE for HEALTH CARE. for even as we continued our TEXAS to disrupt, dismantle, and defeat the FLU SHOTS who drew us into war in the Middle East and North Africa -- words which tell us that the less we use our powers and how we intend to do."	Health Care	● Not particularly coherent ● Noun phrases are not contextually appropriate ● Capital words are replaced NPs

Both attempts had some shortcomings and overall performed worse than our main n-gram Markov chain model. The part of speech templating did not help with coherence and replacing noun phrases sometimes put noun phrases in contexts that did not make much sense.

Noun phrase insertion did seem to go a fairly good job of biasing towards certain topics, so we decided to compare our n-gram Markov model with noun phrase insertion in terms of readability. Both are scored on a 0 to 10 scale with 0 being completely off-topic (for topic score) or unreadable (for readability score) and 10 being completely on-topic or completely readable. We printed out fifteen outputs for each model and topic and used three graders to get an average over the topic and readability score. We then found an overall average of how the model did in general on a topic.

Figure 4: N-gram Markov Model biased towards health care

Generated Text	Topic (0-10)	Readability (0-10)
They will continue to jack up premiums 40 or 50 or 60 percent as they have opportunity. That's what they also know how important it is not. Going forward, we want to divide ourselves along sectarian lines or any of this town to tour the camp, so they don't just need that security, they want to thank all of you well.	7	7.7
The insurance industry will continue to jack up premiums 40 or 50 or 60 percent as they ever were. Our American story continues. It needs to be burdened with massive deficits we did not wage this long political darkness a brighter day will come about from reform -- from medical IT, for example, employers may be unable to find that they are paying premiums for their own lives.	7.3	7
…		
Average:	6.92	6.62

We can then compare this same topic biasing for noun phrase substitution (Figure 5).

Figure 5: Noun Phrase Substitution biased towards health care

Generated Text	Topic (0-10)	Readability (0-10)
Sometimes, the course of Californians, but also as a negative ads serving alongside her introduce generic biologic drugs, congress, small business owners all of our nation has overcome every test before you. And as we and many Iranians wish to come to an lung disease think that it fades into the private insurance companies -- well, love and worship and a people harm – and for our troops so that everybody back home is grateful.	6.7	4.3
With the president's indulgence, I want to welcome him to stay one step -- that those of us -- supporting our ally needs in confronting this challenge, we will launch a 2 billion more in Indonesia, where what began as a nation and remind ourselves that we had a chance to succeed.	2	6.6
..		
Average:	4.4	5.26

For healthcare, the n-gram Markov model averaged a 6.92 topic score and a 6.62 readability score. The noun phrase insertion model averaged a 4.4 topic score and a 5.26 readability score. In this case, our normal n-gram Markov model performed better in both aspects.

Figure 6: N-gram Markov Model biased towards Foreign Policy

Generated Text	Topic (0-10)	Readability (0-10)
But clearly, Iran has influence in Iraq. You are an American Jobs Act. And then, for me to an economic stimulus package that fits your budget, and he is graduating Phi Beta Kappa on his face -- from the outset of the danger of climate change, it's young people who've not yet over.	5	5.7
But clearly, Iran has agreed to create advanced biofuels to power 3 million new jobs. In total, we've lost it a priority and we reduce our warheads and stockpiles, we should be giving Secretary Gates and our two countries is unique. For we have a choice. The Communist regime thought an election captivated the attention of America -- it won't grow.	7.7	7.3
...		
Average:	5.74	6.54

Figure 7: Noun Phrase Substitution biased towards Foreign Policy

Noun Phrase Substitution Foreign Policy	Topic (0-10)	Readability (0-10)
Soon after that triumph, we must help answer the call to come together to provide more access to the task of changing who you are. Question: thank you. Americas, everybody. Today I authorized two limited missions: protecting our recent losses. It's precisely our investments in a weakening of their losses, and pondered the role of government, but were also confronting perhaps the oldest living veterans of laden.	5	6.3
Thank you so much. Thank you so much, everybody. And young daughters. France. Thank you very much, everybody. Terrorist networks overseas you, and then give it our best and brightest to take a few words on a second. they're thinking about those astronauts, and we will have to ensure Iran doesn't get memorialized in a globalized world, we have to recognize that we are going to ask ourselves what is admittedly a very financial centers to the global economy has made Iran with Iran -- through drones or **special forces to give out bags of candy to ISIL**.	7	5
...		
Average:	6.28	5.04

For foreign policy biasing, the n-gram Markov model averaged a 5.74 topic score and a 6.54 readability score. The noun phrase insertion model averaged a 6.28 topic score and a 5.04 readability score. In this case, the noun phrase model performed slightly better in terms of focusing on the topic, but the text suffered in terms of readability which is expected because noun phrase substitution is blind to context.

Overall, the n-gram Markov model seems to be better at combining topic bias and readability, but the performance of noun phrase insertion shows potential for trying to incorporate noun phrases in some manner in future work.

Discussion & Future Analysis

Our research into Natural Language Generation yielded some interesting results, and allowed us to explore several different methods of language analysis and text generation. The main question we were attempting to address was whether or not it would be possible to create a functional text generation system that could take an input of topic and create a coherent and contextually

appropriate political speech. Through a modified Markov text generator with minor high-level templating, we were able to develop a system that could create occasionally readable and somewhat on-topic output. While our speeches probably could not pass for authentic Obama speeches, they provided us with a way to explore the possibilities of language generation, and the strengths and weaknesses of various approaches. We attempted several methods, and found most of them to be deficient in some way when compared to a more simple Markov chain based algorithm. Our attempts to enforce word choice based on part of speech produced output that was incoherent and off-topic due to its focus on filling in the words based on their part of speech tag. Other attempts to bias topics by replacing noun-phrases with noun-phrases extracted from the topical datasets generated outputs that tended to use topically relevant phrases, but in a nonsensical way. This experimentation gave us some insight into the power of the simple Markov chain model, and the way it can be easily modified to generate somewhat readable and presidential sounding speech.

While the greatest challenge was generating coherent and readable speech, biasing speech towards certain topics proved problematic as well. We narrowed our scope to health care and foreign policy, and created datasets of concentrated speech relating to those topics, and weighed our model's word choice using those. Despite this, it was sometimes difficult to keep the speech on topic, as it would not always pick a word from the topic datasets. The unbiased, "vanilla" speech was also sometimes skewed towards a particular topic just based on the bias of subject matter usually covered in Obama's speeches, and the particular speeches our database happened to compile. Through our concentrated dataset approach, we were able to solve the problem of biasing speeches to a certain degree, but the output of the model is not always consistent.

Going forward we believe that our programs could be greatly improved if we implemented more strict rules about how words are chosen. We believed that randomness was important for a true text generator so we allowed our program to randomly select words only on the basis of what words preceded it. We could potentially get into specific details about how long sentences should be, and what general structure they take. Another area which could be developed is punctuation. Right now punctuation is attached to the words when they were scanned in instead of being generated by the program itself. And finally more work could be done to improve the noun

phrase substitution model. The model is interesting because it takes a template like approach as it assumes that noun phrases can be swapped for one another to affect the meaning of a sentence. If we had more control over sentence structure we could create a more readable model.

Ideally we would like to see our Obama speech generator running as a viable candidate for President by 2016.

References

[1] Deemter, Krahmer, and Theune. *Real vs. template-based natural language generation: a false opposition?* http://doc.utwente.nl/65551/1/templates-squib.pdf

[2] Karpathy, Andrej. "The Unreasonable Effectiveness of Recurrent Neural Networks." http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[3] Raag, Shabda. "Generating pseudo random text with Markov chains using Python." http://agiliq.com/blog/2009/06/generating-pseudo-random-text-with-markov-chains-u/