Andrew Paley
Project Status Report

The Task

My task is to take any given news article and automatically extract the quotes (direct and, as a stretch goal, paraphrased) of public figures in it.

As laid out in my project proposal, I believe this task is important as a building blocks towards a system that helps keep public figures accountable for the positions they hold in the public sphere. We're losing a lot of information down the memory hole, and a lot of questionable actors are taking advantage of it. But before we can build systems that can keep track of what individuals in the news are saying, what their statements mean, if their positions have changed, etc, we first have to develop a system that can recognize their quotes to begin with.

So what has Bernie Sanders said about Matt Whitaker's appointment? What did Brian Kemp argue about the voting situation down in Georgia? What does Trump have to say about the slaying of Jamal Khashoggi? Answers to these kinds of questions are the goal.

The Data to Date

The nature of this work is that the majority of it is data collection and preparation. To that end, I've built the first half of my training platform as an ingestion pipeline. It now does the following: 1) searches for daily articles via NewsAPI (based on an initially small list of topics and public figures, defined in the files below); 2) whenever an unknown article shows up, programmatically retrieves the full body of the story (which NewsAPI does not provide for most sources); 3) tags the incoming articles by public figure and topic (from the search queries); 4) stores the articles and their metadata in a coherent way; 5) does coreference resolution; and 6) serves up the results via tooling to review snippets of the stories and label them as quotes or not (my ultimate training data will have to include samples of both quotes and non-quotes, appropriately labeled), before storing them back in the database.

The work is ongoing in this repo: https://github.com/andrewpaley/voicesCarry/ and the key files of interest at the moment are jumbodb.py (a layer on top of SQLite DB for managing articles, topics, public figures and snippets); trunk.py (the class that gets stories from NewsAPI and then follows the links associated with them to scrape the full articles); shepherd.py (a command line interface for extracting snippets via either a) manually reviewing stories or b) reviewing Shepherd's "guesses" based on spaCy's lemmatization features and a manually created set of "verbs of attribution" from online sources); and grok.py (the class that will eventually intelligently guess at quotes -- currently a set of sketches).

I've pulled down 870 articles to date based on a set of topic keywords (Kavanaugh, Mueller, Whitaker, Speaker of the House, and Minimum Wage). I also have scoped the initial list of my

public figures to a narrow set, defined as a constant at the top of the jumbodb.py file linked above.

The Rest of the Path So Far

On top of my burgeoning dataset, I have pursued two lines of exploration, one path leveraging the spaCy NLP library and the other working with Stanford's CoreNLP. Down both paths, I've been in pursuit of a representation that preserves as much of the information from my snippets as possible for the sake of vectorization and then deep learning via Tensorflow. (During the process, the other option that's emerged is training via spaCy's built-in, trainable textCategorizer.)

The big idea: leverage a) coreference resolution (such that shorter sections of text can be analyzed without losing context), and b) dependency tree parsing (such that we can graph the relationships of words within a sentence or passage, as well as identify the role of our public figure of interest in that tree) as a means of reducing the structure of sentences towards something more machine-readable without losing information. My going theory is that there's a classifier for assessing a snippet as a quote (direct or paraphrased) that can be deep learned off of entities and those dependency trees: specifically, the key features would be the entity types themselves, the particular sets of verbs, and the structure of nested clauses (from the dependency tree).

Up Next

1) Iterate on Shepherd to further improve the guessing mechanics and implement the creation and storage of dependency parse trees alongside snippets in JumboDB.
2) Leverage an initial corpus of training data (half/half quotes and non-quotes) to train a model in spaCy's textCategorizer and -- unless I'm wildly successful right off the bat with spaCy alone -- Tensorflow.
3) Possibly add more data now that the pipeline is in place.
4) If successful, build the classifier back into the platform itself as Grok.

Problem Spots/Roadblocks

1) I've given up on CoreNLP after losing a weekend of work to the depths of the library. (Using experimental features while trying to get Python and Java to talk to one another = bad idea. Also, CoreNLP is an enormous resource hog.)
2) In early waves of viewing quotes, I've come to understand that uninformed coreference can be...very uniformed. Without context, statements often don't resolve to the name of the speaker, but instead a context-dependent reference like "the junior senator from Kentucky." This will continue to present problems.
3) In working through ingesting stories, I've come to realize just how varied the styles of quoting people are across publications. Going forward, I'll be scoping my data ingestion to sources from a smaller cache to mitigate the issue (Washington Post, New York Times, USA Today, and a handful of others).