Gray Davidson - Project Fletcher Writeup
5/30/18

Project Design:

In studying NLP I delved into the ideas of unique style and variable word meanings by creating custom word embeddings for a series of different authors. To do this I went through a series of steps. I collected sets of text files for each author (or author group) requiring at least 2+ million words in the source corpus for each embedding. I then loaded these into python and preprocessed them in an automated pipeline. Then I used Gensim's Word2Vec to create the embeddings after experimenting with the hyper-parameters. Armed with the word embeddings I compared the different ways that authors use words. Importantly these embeddings do not represent how they define the words (none of my authors misunderstands the word 'tree') but I do show that in their work these words play different roles from one author to another. Results showed variation in how authors use certain words, ranging from the literal to the utilitarian. Finally I investigated the mathematical viability of the embeddings, finding that indeed, relationships such as [Country >> Capital] were correctly captured.

This project also had several other components which met with mixed success:

— Several non-trivial pre-processing steps
— Implementations of PCA and K-means which did not prove fruitful
— LDA implementation which produced successful but superficial results.
— An attempted embedding on holy texts which was discontinued due to issues with special characters
— Use of an AWS P3-2X instance
— I tested one of three other gensim embedding creators, FastText, which returned similar results (see presentation appendix).

Tools:
The **primary tools** of this project were gensim libraries: (Word2Vec, FastText, Phrases/Phraser, simple_preprocess, remove_stopwords). **Additional tools** were SKlearn's LDA, PCA and K-Means. I **also tackled** AWS, successfully running my models in a Jupyter notebook interface on a P3 instance which I configured according to Chris Albon's tutorial. (shout out to Abed who helped me through some snags in this process).

I will go into greater depth below speaking about **Word2Vec,** **FastText** and **LDA**.

Phrases/Phraser was the tool I used to create bigrams. It is capable of creating trigrams as well but I decided to restrict to bigrams because my corpus was not very large for the applications I was interested in. Additionally I couldn't think of many relevant trigrams within the specific texts I was using. Most were proper nouns such as "Mr. Darcy" or "Middle Earth." I did notice that some erroneous bigrams were created although these did not interfere with my results so I did not chase them down.

I briefly implemented both **PCA** and **K-means** to look at the words in the embedded space but soon decided that more interesting questions could be answered by interrogating the word similarities directly than by clustering.

Finally I spent considerable time getting my code running on an **AWS P3 instance**, working my way through Chris Albon's tutorial so that I could use a python notebook interface on the cloud. I also needed to request permission to use the instance which took some time but by the time I had access I was prepared to run my code start to finish on the more powerful machine. I had previously created a mock dataset which contained snippets of my real dataset and by changing one file path variable I was able to re-run on the full dataset.

Data:
My data for this project consisted of 50+ full texts by a variety of authors. I spent some time searching for corpuses which contained sufficient words to be usable in forming embeddings and which were also freely available online. I ended up combining authors whose work is known to be similar (such as Thoreau and Emerson) to swell several of the corpuses I worked with.

In the future I think it would be fascinating to expand this portion of the project to include, say, twitter data or journalistic texts on foreign policy (really any subset of all written texts which is known for a particular tone or style).

Algorithms:
I used **LDA** to investigate whether the differences between documents were sufficient to ask questions such as "which author wrote this set of words?" or "which of this author's books is this document from?". Here I found success, correctly identifying specific books, and differentiating between authors. I also found that the front matter was conspicuous enough to merit its own 'topic' which aimed specifically at Project Gutenberg and Internet Archive. Interestingly certain 'topics' were formed around character archetypes from within an author's work. Within Rand's books for instance, "Rearden" and "Wynand" formed one topic together. These men are from different works but are both arrogant industrial tycoons and these were correctly identified as related.

The heaviest lifting was done by **Word2Vec** which takes in a list of lists of words (in my case these were preprocessed documents of 100 relevant words from the texts) and creates embeddings for each corpus. Tuning the hyper parameters here led me to look deeper at the preprocessing I had done, including the creation ob bigrams. I elected to use a skip_gram algorithm because it maintains information about the word order as opposed to CBOW which merely looks at _what_ words surrounded a word in question. Skip_gram is preferable for smaller data sets like mine. Normally Word2Vec would be trained on corpuses several orders of magnitude larger than mine but even so I was able to find interesting results.

Interestingly I noticed that gensim **FastText** surfaced many more bigrams than

Word2Vec. FastText was also much more reliant on the structure of a word (i.e. “good” was though to be similar to “Goodwin” and “Goodbye” for this embedding creator as opposed to “better” which would have been more typical of Word2Vec).

Lessons Learned:
What I attempted to do went fairly well in this project. I was happy to meet with no major obstacles in code and to end up with interesting results! The lesson here I believe is that there are always more directions to go and to be judicious in assigning one’s energy. I would like to go forward looking at new datasets, figuring out how to communicate between embeddings and investigating the three other embedding creators in gensim, but this was far too much for a two-week project. I also learned that you should always turn of your P3 instance immediately after using it. :(
#metis/projects/fletcher