Reddit Context Recommender

Chan, Natural

22 March, 2022

Abstract

The goal of this project is to identify Reddit posts that may provide additional relevant context to a new post. We will primarily observe the title of posts from the world news subreddit. From there, we will perform dimensionality reduction and identify key topics within the subreddit. Once we have identified these topics, we will use new posts to identify which posts are the most similar and recommend those to users for further reading.

1 Design

Reddit has grown into a massive social media platform, and has become a major news source for many as well. With new posts constantly being submitted, it is hard to keep track of every single update. This leads to individuals missing context on newer posts when they may have missed a previous post that would have provided relevant information.

Using the titles of Reddit posts, we would like to know: What posts can we recommend to provide context to newer posts? We will answer this question by using matrix decomposition on posts submitted to the world news subreddit. Then, we will obtain newer posts and find the most relevant posts using cosine similarity. Ultimately, this should allow us to recommend related posts and hopefully allow us to find posts that may provide context.

2 Data

The PushShift API saves posts from Reddit, providing a quick way to retrieve posts compared to the Reddit API. We used PMAW, a wrapper for the PushShift API to gather all available posts from the world news Reddit starting from January of 2010 to the present. Unfortunately, the PushShift API does not update non-static information on posts after retrieval. This means that many posts will permanently be recorded with 0-2 upvotes (score) which would result in many erroneous posts in our final corpus. To work around this, we obtained the ID for each post and used PRAW, a wrapper for the Reddit API, to retrieve the upvotes of 100 posts at a time in order to filter out any post with under 1000 upvotes. Once scraped, we had 46408 posts to work with. However, due to PushShift API hardware problems, many older posts are missing so the data is skewed to the more recent years.

3 Algorithms

3.1 Data Cleaning/Preprocessing

To clean our data, we tried multiple tokenization methods with default stop words and then used a bag-of-words method to identify what words are kept. The first method was to apply stemming which yielded good topics for modeling, but also resulted in a loss of some key words. For example, the word "Biden" was stemmed to "bid" which would be a problem due to the frequency of posts regarding Biden in the recent years. Our next attempt was to use lemmatization, but this caused each topic to be dominated by symbols and non-ascii punctuation due to them being labeled as nouns. Using SpaCy, we created a custom lemmatizer which allowed us to remove all non-ascii characters, non-nouns and non-verbs, and other erroneous patterns.

3.2 Modeling

Our primary form of modeling is through matrix decomposition in order to perform topic modeling. We specifically use Non-negative Matrix Factorization over LSA and LDA due to its interpretability, speed on sparse matrices, and effectiveness on short documents. First, we attempted to model our stemmed version of the Document-Term Matrix and searched for 8 topics.

- General Politics
- Russia
- Western Politics
- Trump
- Covid-19
- Protests
- China
- Prison

The topics were pretty decent but there were topics that contained words which may have been better as separate topics. Knowing this, we adjusted to 10 topics when applying our custom tokenizer.

- Death
- Trump/U.S.
- Russia/Ukraine
- Prison

- Asia
- World News
- Hong Kong Protests
- U.K.
- Climate Change
- North Korea

These topics looked much better and using TD-IDF as our vectorizer yielded similar results with slight variations in some. For testing, we used two versions of our model - one using a bag-of-words vectorizer and another using TD-IDF. We chose several recent posts and took a glance at what each model recommended using cosine similarity. For the most part, posts with large relevance in multiple topics did really well at returning similar posts. However posts that were focused on only a few topics had the potential of returning other posts with the same topics despite being unrelated.

Let us take a look at the following post: "Workers Discover 700-Year-Old Lead Coffin Beneath Notre Dame Cathedral". When looking over our topics from earlier, this post didn't seem to fit into a lot of them. Our bag-of-words version recommended the following post: "Shark Numbers Along The Great Barrier Reef Have Dropped By 92% In Just 50 Years". These two posts seem to have no relevance to each other whatsoever - the only word shared between the two posts is "year". Mathematically, this makes sense when we take a look at how cosine similarity is calculated so we adjusted the number of topics to 20.

Now our model recommends the following: "Dozens of new viruses discovered in 15,000-year-old glacier ice". This seems much more relevant so although the 10 topics seemed to be a good representation, it turns out we may want more topics in order to better explain the variance in the posts.

Ultimately, our model did well at recommending other posts when we adjusted the number of topics. However, our goal was to recommend posts that would provide context. This means that we will want a metric that takes magnitude into account which cosine similarity does not do. In the future, we will want to adjust our metric, either by using cosine similarity and then choosing an ϵ value to find neighboring posts, or by creating a new metric altogether.

4 Tools

- PRAW and PMAW to gather data from Reddit
- Pandas for data processing and cleaning
- NLTK and SpaCy for tokenizing corpus

- Sci-kit learn for vectorizing and modeling
- Matplotlib and WordCloud to generate plots

5 Communication

Presentation slides and visualizations are located in my github repository.