Project Title: Reddit Title Generator
Final Project Report

1. Introduction and Problem Statement

One of the most useful techniques when dealing with a large set of text, whether it be in the form of a document or otherwise, is summarization. Text summarization attempts to represent the input text while condensing its form. For our project, we decided to take this idea one step further by creating titles or headlines for our input data. If done properly, a title can act as the essence of the body of text it labels.

Headline generation is the technique of using data from a document (or set of text) to retrieve relevant information and words to create a headline that represents the input text. Headline generation can be done using text summarization techniques or "true" generation by using new words. This project uses a graph-based ranking model to create titles based on comments of posts using the publicly available Reddit comment data set. Evaluation of these titles were done by user studies. We tried two approaches, with the assumption that the second would improve on the first. The first approach was the use of summarization explicitly, by using whole sentences as input and returning the highest ranking sentence as the title. The second approach attempted to be closer to a generator by using a POS template on extracted keywords to form sentences that did not exist in the input data. We found both approaches were preferred fairly evenly in user studies with the former having slightly higher average scores when chosen.

2. Related Work

In the past, common methods of headline generation have been the use of Hidden Markov Models and even TextRank[1]. In fact *How to Compete Online for News Audience: Modeling Words that Attract Clicks* uses TextRank as their baseline for keyword extraction to find common trends in successful words in headlines. Some researchers also implemented template filters to attempt to improve the structure of their generated headlines[3]. Rather than developing a new algorithm, we decided to combine these techniques, using TextRank for summarization initially and then for keyword extraction. However, instead of evaluating trends in headlines, those keywords are used directly on a template structure (the details of which are described later on). We are comparing the results of just the TextRank summarization on whole sentences and TextRank on keyword extraction with a template structure with the assumption the template structure will improve on the results of just summarization.

3. Data Sets

The data used for this project was the publicly available Reddit comment history data set provided by /u/Stuck_In_The_Matrix. This data was ideal for the project as every post had a title and many comments to be used as input. Each execution of title generation would only need one post, so there is a lot of sample data to work with. The entire data set is 1.7 million comments and is 250 GB compressed. The data is structured as a JSON file where each line is a JSON block representing one

comment. The block contains many fields describing the comment such as the subreddit, score, the body of text for the comment and several fields which were important for grouping the comments such as:

Link_id: identified the post the comment came from

Parent_id: identifies the id of the parent comment (is link_id if this comment is the parent)

Id: identifies the comment itself

These identifiers are only unique per subreddit, so the subreddit field was also important for classification purposes. Unfortunately, the data set in its entirety was too large for the scope of this project, so we used a subset: a month of Reddit history (January of 2015 in particular). This data set is about 5 GB compressed (30 GB uncompressed). As mentioned earlier, the project's structure allows for results without the use of a large amount of data, so to ease handling the data, we truncated the one month dataset into 10,000 lines (10,000 comments).

Unfortunately, the data in its raw state did not contain titles in the JSON blocks. The titles would not be used in input but it would eventually be used for comparison throughout the project and in user studies. To get around this, the Python Reddit API was used, which took in a URL formed by using the fields mentioned earlier (ex. reddit.com/r/subreddit/link_id). This would link to the post containing the associated information. By passing this to the Reddit API, HTML requests could be made to extract title information. This information was stored as a dictionary of link_id -> title and saved as a json file for convenient retrieval.

Once the titles were extracted, the truncated data was loaded in and grouped together based on link_id (and subreddit for the reason mentioned earlier). These comments were then stored along with their titles in the format title -> [comments]. This information was also saved as a json file. The resulting data from this operation would be the data we used for the rest of the project.

Note: For the remainder of the report, "data set" will be referring to the truncated data or more specifically the title-> comment groupings that resulted from it

Source for data:
https://www.reddit.com/r/datasets/comments/3bxlg7/i_have_every_publicly_available_reddit_comment/

Visualization of data (credit to Felipe Hoffa): https://www.youtube.com/watch?v=l8MLIfU21pk
Note: This visualization is for the whole data set (250 GB)

4. Description of Technical Approach

This project was split into two phases, each of which explored a different approach on headline generation.

Phase One

The first phase was a naïve solution to the title generation acting as not only a baseline but an opportunity to gain experience with the algorithm that was used. TextRank was the algorithm chosen to create the titles. TextRank is a graph based algorithm which, when used for summarization, takes in sentences (in this case the comments). The sentences are extracted and a graph is built where every sentence is a node and links are made from each node to all other nodes based on a similarity function. The implementation of TextRank used for this project uses Levenshtein distance as the similarity function which is the minimum number of single-character edits required to change one word into another. Once the graph is constructed, PageRank is ran on it. PageRank is the algorithm created by one of Google's co-founders, Larry Page. PageRank simulates random walks over the graph and in this case based on the weights formed earlier, randomly selects a neighbor. A score is kept and increased for each node each time it is visited during the walks. The sentences can then be sorted based on these scores and the top-n can be used as a summary. The naïve solution used TextRank on the comments and chose the top sentence returned by the algorithm as the title.

Phase Two

In the second phase, TextRank was used again, except this time to extract keywords (instead of entire sentences). The keywords were clustered into phrases to preserve the text content of the input data[2]. The phrases were then POS-tagged. The order of the phrases being returned are maintained as they are returned from most important to least important by the TextRank algorithm. The phrases are then stored into a dictionary based on their POS tag (maintaining relative ordering of each tag). The phrases were then presented to a template filter[3]. The format of the template was decided from some experimentation on the data set. Initially, each title was tokenized and POS-tagged, these tags were then joined by "/" to create a structure. A count was kept of each structure and each time a title had this structure, the count was incremented. The thought process was to find the most commonly occurring structure for use as a template. Unfortunately, we found that the Reddit titles had too unique a structure, as is to be expected from text constructed by humans. We then instead counted the occurrence of every POS-tag in titles of the dataset instead of its structure. The results showed that nouns, pronouns, verbs and adjectives had the highest occurrences. This knowledge was combined with the idea that a common structure for English sentences is "Subject + Verb + Object" to create a template composed of "Subject + Verb + Adjective + Object" where the subject prefers pronouns and the object prefers nouns. The word at the front of the list of its respective POS category in the dictionary is chosen to be filled in the template (as it's the highest rank for that category). The keyword is removed from the list so if a POS reoccurs in the template (e.g. Subject and Object) then the next highest keyword in that category is chosen. The chosen keywords are concatenated and returned as the title. The expectation of this method was to create grammatically correct sentences while making new combinations of words that did not exist in the input data. The template sentences were then presented along with their baseline equivalent in random order to be evaluated in user studies (described in greater detail in Experiments and Evaluation).

5. Software

Below is a table summarizing code that was publicly available along with functions and scripts that were written by us followed by some brief explanations of their functionality.

Publicly-Available Code	Code Written/Tested
Programming Languages: • Python 3.5 and libraries such as NLTK, json, and PRAW Keyword Extraction: • TextRank[4]	Data Parsing Scripts • Used PRAW to grab titles from truncated data • Organized dataset into grouping of extracted titles and grouped comments together for ease of use during experimentation Ranking Algorithm: • Modify TextRank to extract sentences • Evaluate top sentences choosing the topmost Phrase Clustering: • Cluster keywords extracted (forming useful n-grams) Title Generation: • Create a template • Order keywords based on importance and group by POS tag • Use template to generate grammatically accurate sentences with meaningful phrases

TextRank is a core algorithm in this project. Our initial understanding of the algorithm was limited - we opted to use and learn from the publicly available implementation written by David Adamo [4]. As stated earlier, it creates a graph where the nodes are words/sentences and weights the edges based on the chosen similarity function then runs PageRank on the graph. The input here was the list of comments which were tokenized to represent the nodes. The output was the resulting sentences/words with their associated rank.

The data parsing scripts were used to explore the initial truncated data set to extract titles and associate comments with each of them for ease of use in the rest of the project (as explained in Data Sets).

Phase Two generation worked in a similar style to the initial use of TextRank except this time the nodes were words and the results were manipulated further by the use of templates which we made based on our observations (as explained in Technical Approach).

6. Experiments and Evaluation

Initially, we planned on using both user studies and ROUGE for evaluation methods. As we moved further along with the project, we realized that ROUGE would prove difficult given the output of title generation. In the baseline, only a single sentence is returned after summarization is performed. ROUGE would normally evaluate the quality of the summarization based on input data but the technique in the

baseline is not a summarization. As for the second method of using template for title generation, ROUGE was even less likely of a choice as this approach does not even use summarization for sentence creation and instead uses keywords to create new sentences.

As a result, we decided to go with just user studies as our method of evaluation. User studies has many perks in this context as a form of evaluation. The metric in which to set up the user study would be up to us instead of some preconfigured algorithm, allowing for us to test in a way that would be flexible for comparing two different approaches to the same task.

For the remainder of this report, Method 1 will refer to the baseline approach and Method 2 will be used for the template-sentence approach.

When creating our user study sheet, we wanted to measure in a way where Method 1 and 2 could be evaluated simultaneously while providing insight on each separately and directly in comparison to each other. In the beginning, we wanted to compare each method to the original title but after obtaining results, neither method produced titles believable enough to trick a user. To work around this, we presented our recipients with 5 pairs of generated titles, with the original title listed above each pair. The order of appearance of Method 1 or 2 was randomized and the generated titles were capitalized as to not create bias based on proper capitalization (as Method 1 would contain more properly capitalized sentences being made by people). The two generated titles were labeled Title 1 and Title 2 (not indicative of method as mentioned earlier) and the user was asked to choose which title they prefer and also to provide a score from 1 to 10, where 1 is a terrible title and 10 seems like a reasonable replacement for the original title. The scoring system was used in hopes of finding trends in effectiveness between Method 1 and 2 even if one is clearly favored over the other.

Here are the results of the user study (the group size was arbitrary):

User #	**Preference Rate**		**Average Score**	
	Method 1	Method 2	Method 1	Method 2
1	60%	40%	7	5
2	20%	80%	8	7.5
3	60%	40%	9	6.5
4	60%	40%	4.67	2
5	40%	60%	6.5	6
6	60%	40%	6	5.5
7	80%	20%	8	7

8	20%	80%	4	5.75
TOTAL	**50%**	**50%**	**6.85**	**5.8**

In the table, for the Preference Rate section, there is a percentage denoting how often the user preferred each method for the 5 pairs of titles. Given that there were only 5 pairs, the percentages only differs in steps of 20%, however they still show insightful information on user's reactions which we will discuss shortly. As mentioned earlier, the user was also asked to rank their selected title on a scale from 1 to 10. These scores were also taken into account and averaged for each user. Both the preference rate and scores were accumulated to find the overall preference rate as well as the overall average score for each method, this information is listed in the Total section

Although the degree of acceptance can vary greatly from user to user, looking at the Total results, it seems that the preference between the two methods was 50/50. However, on average, Method 1 scored higher when chosen as the preferred title. This result makes sense as Method 1 uses human-written sentences as output. Although these titles are very long (since TextRank summarization favored long sentences), they were more readable and relatable to the users.

We designed this user study with the thought that it was the best test for representation of the two methods and their effectiveness, however, it does contain some flaws. For example, it could be argued that 5 titles may be too small a sample size but we believe a relatively small test would keep the user engaged/focused and reduce the probability of them giving 'insincere' scores, perhaps to finish faster. (Our users consisted of students stressing about their own classes). Another issue is the score itself, a ranking ranging from 1 to 10 is fairly standard but the degree and expectation of what belongs in a 1 or a 10 may be wildly different for each user (indicated in the huge difference of average score per user). We attempted to mitigate this by averaging all scores for each method. No evaluation method is perfect but we tried to utilize the strengths of user studies while attenuating its shortcomings.

7. Discussion and Conclusion

This project was quite a learning experience. Although our methods were not as effective as anticipated, we gained great insight into what makes title generation effective and how difficult it can be to generate titles that can mirror those made by people. Also, in the beginning, TextRank felt like a black box but after using it in both of methods we gained a complete understanding of the algorithm as well as a deeper appreciation for its core part, PageRank, in practical situations (e.g. ranking webpages).

Often times in projects of this nature, there are many things that can occur that are surprising or unexpected. The results of our user study were one such outcome as it managed to both surprise us and somewhat meet our expectation. Initially, during the drafting of the proposal, we had expected Method 1 to be fairly poor and perhaps not even usable as a baseline and that Method 2 would be a significant improvement on those results. These expectations seemed to be reinforced after Method 1's completion as it did not seem to be believable as a title replacement. Once we moved on to Method 2, we believed that it may have not been as good as we initially thought since the template would be a constraining factor on randomness and believability of the generated sentences. After reviewing the

results of Method 2, the expectation was flipped entirely as the template did constrain results more than we hoped and it was assumed that Method 1 would win by a landslide during user studies. What surprised us was the average preference rate between the two methods being 50/50, this was unlike either of our expectations throughout the project. We believe that this may have been caused from the volatile nature of our generated titles. With our algorithm, if a generated title is 'good', it is completely usable as a title and captures the essence of the original title - but when it's a 'bad' title, it can seem random and make no sense within the context of the original title. There did not seem to be a middle ground. This is also reflected in the individual preference rates where some users lean heavily on one method and some the other. What somewhat met our expectations was the resulting total average scores between the two methods. Even though the preference rate between the two methods was even, Method 1 tended to receive a higher score on average when chosen. This is most likely a result of what we mentioned earlier of Method 1 returning titles that are the derived from human-written comments and as such, more relatable to the user.

There are also some benefits and limitations to this problem compared to other approaches that we feel are important to discuss, as any project should attempt an objective approach when criticizing their own work. We believe Markov Chains to be one of the most effective methods in creating believable titles. However, for a Markov Chain you would need a fairly large corpus and a deep understanding of this data to create the states for its use. Although PageRank may be viewed and understood as a Markov chain itself, our method is able to create titles with very little input data (one post's worth of comments). The tradeoff here is the titles of our method being less effective in comparison.

The benefit of using the Reddit data set was that it was easy to handle and group into comments for the small input we needed. However, Reddit threads are posts with users responding to it. This meant that our methods depended on the comments being relevant responses to the title while also being representative of that title, which is why some of our generated titles seem more like answers. There's also a dissonance in source because the person writing the original title and post will not be the same as those who are commenting to respond to it. Each person may have their own tone and degree of relevance and it's difficult to capture that in a title.

If we had a full team working on the problem of headline generation, we could foresee two options depending on our approach. If we decided to stick to our approach of the use of TextRank and dealing with data sets such as Reddit, then it may be in our best interest to give unique weight to comments based on their upvotes and maybe hierarchy in a comment chain (parent comment being the top). This would assume that top comments or those ranked higher by other users may have more relevance to the post. If we went with an entirely different approach then it would be interesting to see how Markov Chains could be applied and optimized on the Reddit data set to create realistic headlines. In fact, something like this already exists for Reddit called /r/SubredditSimulator which is a subreddit where all the posts and comments in them are made by bots, all generated by Markov Chains[6]. Overall, headline and even text generation in general still has a lot of potential and it will be interesting to see how it develops as time goes on.

References and Links

[1] Kim, Joon Hee, et. al. “How to Compete Online for News Audience: Modeling Words That Attract Clicks.” N.p., n.d. Web. < http://www.kdd.org/kdd2016/papers/files/rpp1054-kimAemb.pdf>.

[2] Xu, Songhua. "Keyword Extraction and Headline Generation Using Novel Word Features." N.p., n.d. Web. <https://pdfs.semanticscholar.org/b6ad/9ff01b7fffa2c02718e24548c220b5da2ece.pdf>.

[3] Liang Zhou and Eduard Hovy. “Template-Filtered Headline Summarization.” N.p., n.d. Web. < http://www.isi.edu/natural-language/people/hovy/papers/04ACL-ws-headline-summ.pdf>.

[4] David Adamo. “Python Implementation of TextRank algorithm”. Web. <https://github.com/davidadamojr/TextRank>

[5] https://www.quora.com/What-is-a-simple-but-detailed-explanation-of-Textrank

[6] https://www.reddit.com/r/SubredditSimulator/