Social Media Analysis of President Candidates in USA

Chu Pak Sum
Department of Computer Science and Engineering
The Chinese University of Hong Kong
1155068765@link.cuhk.edu.hk

Filip Strycko
Department of Computer Science and Engineering
The Chinese University of Hong Kong
filip.strycko@link.cuhk.edu.hk

Niclas Ogeryd
Department of Computer Science and Engineering
The Chinese University of Hong Kong
1155076043@link.cuhk.edu.hk

Zhao Che
Department of Computer Science and Engineering
The Chinese University of Hong Kong
1155066626@link.cuhk.edu.hk

ABSTRACT

The goal of this paper is to analyze content shared in social media by presidential candidates in the United States of America. In particular, Twitter posts are being inspected, using advanced methods of big data analysis. In this report we describe the usage of tools, such as Map Reduce, collaborative filtering, k-means clustering and others, to determine different features of candidates' communication. We also made an application for recommending ideas to candidates based on the their Twitter analysis. At the end we show overview of our findings and propose directions for further analysis.

Keywords

Twitter analysis; USA president election; natural language processing; topic extraction; map reduce; recommender system

1. INTRODUCTION

With increasing popularity of social networks, social media are becoming more and more important as communication channel between well known person and the public. Actors, politicians and other celebrities are investing into marketing and public relations to form their identity in professional manner.

With upcoming presidential elections in the United States of America, candidates are heavily using social media to communicate with potential voters and putting a lot of efforts into crafting their posts, comments and other content. In this paper we analyze social media content (particularly, Twitter) of candidates to compare different features of the content they release. In particular, we want to analyze posts on the personal account pages of candidates. We want to find frequent words occurring in their posts. Also, we want to extract important keywords from the posts and use them to find out the topic of particular posts. Further we can compare posts with the same topics for their similarity, study popularity of different topics or use other statistical methods to analyze the results.

Presidential elections in United States of America are scheduled on 8th November, 2016. Many of the candidates use Twitter to build image of themselves and to communicate with their current or potential supporters. In this elections, there is large number of candidates and analyzing all of them would not be possible within tight time boundaries of this course. However, we chose seven candidates, which according to our brief research will attract the most attention and influence the results of the election in the largest scale.

We chose following candidates:

- Jeb Bush
- Ben Carson
- Hillary Clinton
- Ted Cruz
- Rand Paul
- Marco Rubio
- Donald Trump

2. DATA OBTAINING AND PRE-PROCESSING

Twitter's public REST API provides a convenient way to obtain different kinds of data from Twitter. We mainly focus on the information of posts (or tweets) from the selected candidates and they can be obtained using the GET statuses/user_timeline method in the API. The method will return a JSON file containing a collection of tweets information of a particular candidate, which contain a lot of features of tweets. As the there are seven selected candidates, we obtain seven JSON files in all.

Here is a short piece of a JSON file returned by the method:

```
...
"text": "Introducing the Twitter Certified
         Products Program: https://t.co/MjJ8xAnT
         ",
"retweet_count": 121,
```

```
"in_reply_to_status_id_str": null,
"id": 240859602684612608,
"geo": null,
"retweeted": false,
"possibly_sensitive": false,
"in_reply_to_user_id": null,
"place": null,
...
```


In this project, we obtained all posts of the candidates from 2014 to now, which are around twenty thousand in total. Besides, we only used five tweet features that we thought are the most important. They are:

- id
- text
- favorite_count (number of likes)
- retweet_count (number of retweets)
- created_at (tweeted time)

We did pre-processing to extract these features of tweets from every JSON file and stored them to seven CSV files. This task was completed easily by using a Python Twitter API wrapper named python-twitter[6]
Here is one example record from one of the CSV files we finally got:

661333253115236352, Grateful to spend time today with mothers who have lost a child to violence and turned their grief into a national call to action. -H, 1365, 563, Tue Nov 03 00:05:09 +0000 2015

3. TOPIC EXTRACTION OF TWEETS

In order to get more meaningful information from tweets, we should find a way to know what every candidate is talking about on Twitter, and what kind of issues his or her followers are most interested in. Thus we tried to extract topics from the text of every tweet, which is mainly a natural language processing(NLP) task. In this project, we used a popular Python NLP library named Natural Language Toolkit[1] for this task.
"NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming/lemmatization, tagging, parsing, and semantic reasoning." [1]

3.1 Definition of Topic

The definition of topics of a tweet is an interesting issue. We thought that there were two main kinds of definition: singular word and multiple words (phrase) from the tweet text.
Using singular words in the text may be the most direct way for topic extraction, but we should have more specific rules to extract only those meaningful words. Simply using all the words will obviously lead to bad outputs. Singular words may not have the ability to represent a more specific topic, so we also took phrases into consideration.

3.2 Word as Topic

To extract words as topics, firstly we used a tokenizer method in NLTK to segment words in the text data retrieved from Twitter. All the punctuation and URLs in the text were discarded in this step. Secondly we did further filtering of the words, discarding all the digits, words that have length of less than three and words that are common stop words in English.
Then we formatted all the remained words into lowercase. Then we tagged words using the tag set in the Penn Treebank Project[4].
Finally we used stemming/lemmatization techniques to format inflected (or sometimes derived) words.
Here is some explanations of stemming and lemmatization: "stemming usually refers to a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes. Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma"[5].
Stemming is faster than lemmatization, but the quality of lemmatization is relatively better. NLTK has implemented functions for both of them. In our experiments, we tried SnowballStemmer for stemming and WordNetLemmatizer for lemmatization. At last, we chose lemmatization to ensure better outcomes.
Table1 shows some samples of word topics from Hillary Clinton's tweets. Each row represents a word of a tweet, with lemma, numbers of favorite and numbers of retweet follow.

lemma	favorite	retweet	word	tag
grateful	1365	563	Grateful	adjective
grief	1365	563	grief	adjective
gaining	0	123	gaining	gerund
quality	0	123	quality	noun
combined	833	665	combined	past participle

Table 1: Word Topic Samples of Hillary Clinton

We can see that in **Table 1** words like "grateful", "gaining", and "combined" were extracted as topics. Word like them do have some meanings, but are not much like the topics we are talking about. So we did another experiment to extract only nouns in the text to be topics. They are words that have tags "noun", "plural noun", "proper noun" or "proper plural noun".
Table 2 shows some samples of noun topics from Hillary Clinton's tweets. Each row represents a noun word in one post, its tag and lemma, and the favorite count and retweet count of the tweet it belongs to.

lemma	favorite	retweet	noun	tag
spend	1365	563	spend	noun
mother	1365	563	mothers	noun
violence	1365	563	violence	noun
quality	0	123	quality	noun
manager	833	665	managers	plural noun

Table 2: Noun Topic Samples of Hillary Clinton

The result seems better than using words with all kinds of tags.

3.3 Phrase as Topic

Collocations are expressions of multiple words which commonly co-occur[3]. We can see them as phrases in the corpus. To extract collocations, we followed the same works as **3.2** to tokenize words, remove meaningless words and other characters and format the remaining words in lowercase. Then we applied pointwise mutual information (as **Figure 1**) with the BigramCollocationFinder function and the TrigramCollocationFinder function in NLTK to extract bigram collocations and trigram collocations from all the text corpus from each candidate's Twitter timeline. Finally we rank collocations by PMI.

$$\mathrm{pmi}(x;y) \equiv \log \frac{p(x,y)}{p(x)p(y)} = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)}.$$

Figure 1: Pointwise Mutual Information

Here are the results of the top bigram and trigram Collocation topics of Hillary Clinton and Jeb Bush.

Hillary Clinton	Jeb Bush
hillary plan	iran deal
women rights	campaign trail
rights act	choice program
voting rights	thanks coming
climate change	big announcement
hillary fight	the president
hillary snapchat	conservative reforms
immigration reform	school choice
gun violence	looking forward
human rights	new hampshire

Table 3: Top 10 Bigram Collocation Topics

Hillary Clinton	Jeb Bush
dinner with hillary	reform growth plan
watch live hillary	health care plan
happy mother day	credit scholarship program
equal pay paid	must work together
defund planned parenthood	million new jobs
hillary enter spot	tax plan puts
hillary new hampshire	terrible iran deal
live hillary speaks	had great time
women rights human	thanks everyone came
ahead america gets	jeb bush tax

Table 4: Top 10 Trigram Collocation Topics

We can see in **Table 3** and **Table 4** that some topics did make much sense. There is a big style difference between topics of Hillary Clinton and Jeb Bush, and most of the main issues they concerned about are not the same. And it is also interesting that both of them are quite concerned about the New Hampshire. There is an obvious further improvement we can do: topics containing the name of the candidate politician like "Hillary" can be removed.

3.4 Clustering of Noun Topic Samples

Another method we were experimenting with was clustering. We assumed that if we can extract sufficient topic representative keywords from the post, two posts with the similar keywords will fall into the same cluster. In this method we considered Noun Topic Samples as the representative keywords, as that was the best what we could withing project's time constraints get.

For clustering we decided to use our own implementation of *k-means* clustering algorithm. At first, we used MinHashing technique for creating binary vectors for each post. We created total set of all keywords extracted through the candidate's posts, and then for each post we created binary vector indicating which keywords the post contains. Binary vector of each post has size s, where $s = |S|$ and S is the set of all keywords extracted from posts of particular candidate. For measuring similarity of two post binary vectors, we used *Jaccard similarity*.

Throughout the whole algorithm we were using tuples of post binary vector and actual set of keywords of the post. At the end of clustering, this set of keywords helped us to describe the clusters and interpret it with the topic.

In our project this method turned not to provide sufficient results. In theory, this method should provide decent results, however, the keywords extracted from the topic must really precisely reflect meaning of the post. In our case we couldn't get the sufficient keyword extraction algorithm in the time of this project.

Another drawback of this method is caused by basic theory of the algorithm. Basically it just finds entities which are the most similar to each other. And at the end it doesn't describe concrete clusters themselves. Thus, it is necessary to interpret topic of the clusters at the end of the algorithm manually, be considering keywords in the clusters. Of course, another option is to make it with some another algorithm.

Using this technique is also relatively time consuming, as it contains number of a priori parameters we needed to set. We spend a lot of time just trying different values and analyzing data.

One of the ideas how this technique could be improved, is to use weights on the keywords, considering different groups (nouns, verbs ...).

3.5 Conclusion

From our experiments, we drew a conclusion that using phrases (collocations) as topics may be the relevantly better method to extract topics. However, due to a limitation of time, we only used word topics in later analyses and applications.

4. MAP REDUCE

We used the Map Reduce algorithm to find words or topics with different features. As input we took pre-processed data, which contained either keywords or topics we retrieved from the posts. This process is described in **Section 3** of this paper.

In this project we used *Hadoop streaming* technology, which allowed us to use Python programming language, which we used also in all tasks across this project. We use Map Reduce technique to achieve three particular goals:

- What are the most frequently used words/topics of the candidate
- What are the most popular words/topics of the candidate
- How big mass of people can candidate attract with his or her post

Solution was designed in the way, that procedure of processing data was the same in all goals. At the end, we sort the output according to our needs, to obtain order of keywords/topics according to different parameters.

4.1 Input data

Data used as an input for mapper were pre-processed separately, so mapper didn't have to do all the natural language processing, which would be very time consuming on the Amazon machine we used for Map reduce. When using the Mapper, it doesn't matter if *keyword based data* or *topic based data* are used as an input, because processing is preformed in the same way for both cases. If the input data are *keyword based*, each line of the input represents keyword of the twitter post, with particular information from the whole post. For example:

mother 1365 563 mothers NNS hillaryclinton Tue Nov 03 00:05:09 +0000 2015

says that keyword "mother" was found in tweet which was liked 1365 times, retweeted 536 times, original word is "mothers", it's noun in plural (NNS), belongs to twitter name "hillaryclinton" and was published on particular date.
If the input data are *topic based*, each line represents topic retrieved from the tweet and the parameters of that particular tweet. For example:

women rights 1365 563 mothers NNS hillaryclinton Tue Nov 03 00:05:09 +0000 2015

Where "women rights" is topic of the tweet and the rest of parameters are the same as in *keyword based* input. In the rest of this section the name **token** will be used to represent either keyword or topic of the input.
We would like to stress the importance of pre-processing, as the crucial part of analysis. The Map-reduce part of the solution will only compute parameters and occurrences of the tokens. Therefore, when the input topic is retrieved, it doesn't represent the real meaning of the post. Similarily, if a keyword doesn't correspond to the semantic of the post, Map-reduce will never return the reasonable output. Thus, the input provided in this project can be just as good as the output of the Map-reduce program.

4.2 Most frequent words / topics used

To find out the most frequent words or topics used by the candidate, we had to count occurrence of each token in the input. After using Map reduce, just sorting of the output is needed to obtain the right data set. This task is very similar to the widely known *word count* example of Map reduce algorithm, therefore it's not necessary to go into details.

Candidate	Words
Jeb Bush	Great, Thanks, Today, New, Florida, Day
Ben Carson	RealBenCarson, Thank, Tonight, Book, OneNation, Sign
Hillary Clinton	HillaryClinton, Women, Rights, Today, Family, American
Ted Cruz	Join, CruzCrew, Today, CruzCountry, America, President
Rand Paul	StandWithRand, Today, Join, Campain, Tax, Debate
Marco Rubio	Click, American, Today, Watch, Century, Thanks
Donald Trump	realDonaldTrump, Great, Poll, America, MakeAmericaGreatAgain, People

Table 5: Most frequent words

4.3 Most popular words / topics used

To found out most popular tokens, we had to define what popularity means. We have different parameters to consider, such as number of people who liked (we say that person 'likes' the post) the post, number of retweets and number of occurrences of the token. At the beginning we used a simple formula:

$$popularity = \frac{likes + 1.5 \times retweets}{occurrences}$$

This equation considers likes, retweets and occurrences of the token. Retweet value is multiplied by co-efficient, because it weight is from our observation higher than the like. In other words, if somebody 'retweets' the post, it matters more than if the person 'likes' it. Naturally, we can say that post is popular when people like it or retweet it. Tokens mentioned more often have therefore higher chances to be popular, therefore we have to bring influence of number of occurrences of the token to the formula. Sum of likes and weighted retweets is therefore divided by number of occurrences.
However, during the works on the project, we figured out that this formula is not the optimal. The example of it is post of Hillary Clinton:

It's so much more fun to watch FOX when it's someone else being blitzed & sacked! #SuperBowl

This post has 54 079 retweets and 41 254 likes and according to our system, post is about "superbowl" and receives a very big popularity index, because it is the only post that our system evaluated as post about "superbowl". Hillary Clinton is known for her fights for gender equality and women's rights in many areas. She has a lot of posts mentioning this topic and gets a lot of attraction and popularity among her supporters and on twitter as well. However, our system gives "women" token lower popularity index then the index "superbowl" received.
Our systems returns reasonable output regarding popularity and reflects some facts about the candidate. However, if we had more time for this project, this would be the area where we should spend more of it.
After obtaining popularity index, the output is sorted by

Candidate	MAI	Ratio
Jeb Bush	32 775	0,0233
Ben Carson	85 792	0,0612
Hillary Clinton	686 192	0,4899
Ted Cruz	76 275	0,0545
Rand Paul	108 161	0,0772
Marco Rubio	41 680	0,0298
Donald Trump	1 400 663	1

Table 7: Mass Attraction Index

this metric in order to transform the data to the format that we desire.

Candidate	Words
Jeb Bush	Christ, Sorry, Pontifex, Fixed, Possibility, Explore
Ben Carson	Quitude, IAmAChristian, Lord, Voice, Yes, Listen
Hillary Clinton	Superbowl, VaccinesWork, Boy, Planet, Science, BringBackOurGirls
Ted Cruz	Determined, Pluto, NASANewHorizons Juntos, RETWEET, Discover
Rand Paul	Renewal, Uber, Lift, Reality, Successful, BarnieSanders
Marco Rubio	Cup, USWNT, Champions, SheBelieves, ussoccer_wnt, FINAL
Donald Trump	Mentioned, Fence, Notice, Tuition, Religious, Wasting

Table 6: Most popular words

4.4 Mass Attraction Index (MAI)

To determine how popular the candidates are, we used the value of popularity of the strongest topic of the candidate. In other words, the largest popularity candidate was able to achieve with some topic. We assume that this value indicates the strength of the fan base of the candidate. In the **Table 7** we also included it's comparison with the largest value among the candidates, as "ratio"

5. POPULAR TOPICS RECOMMENDER SYSTEM

5.1 Motivation

According to the finding of social media analysis, we found that politicians always talk about what the public concerned most, and always want to know what the public like. With the help of natural language processing technology and collaborative filtering. It is now possible to suggest popular topics to a candidate or any social media user what their visitors interested most.

5.2 Introduction

"Popular Topics Recommender System" analyzes the Twitter posts of a user, and then suggests popular topics which are related to the ideas that the user is going to write. With the "Topic Popularity Rating" which is estimated by the number of likes and retweets, this system can collaborate the tweets together and find out the hidden relationships between topics.
From politicians' posts to visitors' likes, retweets and responses, it can be thousands of messages everyday. Which is a big data problem if a candidate want to find out what did people concern for a certain period of time.

5.3 Design

"Popular Topics Recommender System" suggests topics to Twitter users by collaborative filtering, which is based on doing singular vector decomposition on the Rating Matrix as shown in **Table 8**. In the matrix we have columns for the topics, and rows for the corresponding "Topic Popularity Rating" of every tweet. By manipulating this matrix, the system can be trained by the posts, and can suggest topics to the ideas.

Table 8: Lemma Rating Matrix for Collaborative Filtering

	politicianf	pay	decision	woman	time
Post 1	3.124	3.151			
Post 2		4.622	4.637		4.602
Post 3		1.295	1.254		
Post 4				0.501	
...					
Idea 1				4.00	4.50
Idea 2	3.50	3.00			

In order to generate topics for the matrix, we extract topics by applying the topic extraction techniques we have used for Twitter analysis. And defined a simplified "Topic Popularity Rating" formula for the numbers. Both of them will be describe in detail in the coming sections.
As "a week is a long time for politics"[7], we have to prevent feeding the system out-dated posts. But one week of tweets won't be enough for training the system, so we decided to feed the system the latest thirty days tweets.

5.3.1 Development Tools

This system was implemented in Python programming language, and made use of "NLTK" for topic extraction, "Crab" as our Recommender System Engine.
"Crab, as known as Scikits.recommender is a Python framework for building recommender engines integrated with the world of scientific Python packages (numpy, scipy, matplotlib). The Crab engine aims to provide a rich set of components from which you can construct a customized recommender system from a set of algorithms and be usable in various contexts such as science and engineering."[2]

5.3.2 Extract Topics from Twitter Posts

Topics are extracted with the techniques that has been introduced in **Section 3**. For simplicity reasons, "Popular Topic Recommender System" only finds topics from the lemmas of nouns. Below is an example for showing the extraction process in five steps.

Take the following Twitter post as an example:

"RT @TheBriefing2016: Hillary is fighting for equal pay for women. Republicans are...not. #GOPDebate"

Step 1: Remove URL from the content
Step 2: Remove all punctuation marks and numbers
Step 3: Remove all stop words like the, is, am, are etc
Step 4: Extract nouns from the post and convert to lowercase

- women
- pay
- gopdebate
- thebriefing2016

Step 5: Convert plural nouns to their singular lemmas

- women => woman
- pay => pay
- gopdebate => gopdebate
- thebriefing2016 => thebriefing2016

Finally, we extract "woman", "pay", "gopdebate" and "thebriefing2016" as the post's topics

5.3.3 Topic Popularity Rating

For simplicity's sake, the combination of the number of likes and the number of retweets has been used for the rating. Retweets are weighted higher as re-posting can make more influence than likes. As a post of a popular twitter account can have over ten thousands of likes and retweets, it has to take a log10 to the rating for keeping the number within an acceptable small range for calculation.

$$R_{popularity} = log_{10}(LIKES + 1.5 \times RETWEETS)$$

For collaborative filtering, some random noise should be added to the "Topic Popularity Rating". The noise should be very small comparing with the rating, otherwise it will affect the accuracy of the recommendations. Thus, on top of the popularity rating equation, a noise term has been added as our final popularity rating:

$$R_{popularity} = log_{10}(LIKES+1.5\times RETWEETS)+random(0.01)$$

Figure 2: Hillary Clinton Twitter Post

Take the Hillary Post of **Figure 2** as an example, there are 4 091 retweets and 7 243 likes, its Topic Popularity Rating is 4.126 + 0.003 = 4.129.

5.3.4 Collaborative Filtering

With both the extracted topic and popularity rating in place, a topic rating matrix similar to the example at **Table 8** can be created for collaborative filtering. The recommender system engine, Crab, will find out the relationships of topics by doing singular vector decomposition on the matrix. In fact, the noise added to the rating can ensure the matrix is invertible for SVD.

5.4 Validation and Evaluation

The accuracy of recommendations was evaluated by making recommendations according to the lemma of posts of Hilary Clinton's twitter posts. The recommendations were then cross checked manually with the posts to see if they are meaningful.
To our team, around 60% of the recommendations are useful. It's worth to note that US people and the candidates should find the suggestions more understandable than us. It is because they know better about the American culture.

5.4.1 Examples of Popular Topics Recommendation

Below are good and bad recommendation examples from Hillary Clinton's Twitter. More examples are available in **Table 9**.

Good Recommendations

Twitter Post:

We can't wait any longer to keep our communities safe. We need to act on gun violence prevention. #GOPdebate

Ideas:

From the posts we got the following topics (or lemmas) as ideas:

- prevention
- community
- gopdebate
- violence

Recommendations:

Popular Topics Recommender System gave the the following suggestions:

Recommended Topic	Popularity Rating
act	3.691
demdebate	3.666
amendment	3.650
family	3.511
life	3.494
measure	3.493
president	3.239

Where life and measure are closely related to violence; amendment and measure are related to prevention; president is related to is GOP debate. With these suggesion Hillary should possible to elaborate her ideas where the public concern easily.

Table 9: Example of Recommendations

Ideas	Recommended Lemmas
Wall Street, behavior, plan	jail, crime, commit, work, prosperity, accountability, regulation, crisis
Americian, health, care	politician, decision, woman, card, pay, deal, parenthood, affordability, quality
birthday, today	parenthood, Hillary
president, Obama	Barack, accomplishment, senior, secretary, United State, advisor
option, world, retreat, responsibility	tragedy, diplomat, place
secretary, fellow, tenure, state, power	DEM debate, Obama, president, video, America, world, place
family, community, violence, president	amendment, DEM debate, gun, time, NRA: National Rifle Association,
	KeepShoutingOnGuns: Twitter account, prevention, GOP debate, mission, income
child, way, life, adult	ImWithHer: Twitter account, woman
violence, policy, prevention	NYDailyNews: Twitter account, courage, gun, plan

Bad Recommendations

Twitter post:

@katyperry, you bring the campaign dress code to the next level. Thanks for hanging out with us in Iowa! -H

Ideas:

From the posts we got the following topics (or lemmas) as ideas:

- iowa
- dress
- campaign
- code
- level
- thanks

Recommendations:

Popular Topics Recommender System gave the the following suggestions:

Recommended Topic	Popularity Rating
billclinton	2.947
day	2.947
hillyesm	2.946

This recommendation is bad from our team's perpective. The message should be meaningful between @Hillary and @katyperry, but which is hard for us to understand.

6. ANALYSIS

6.1 Frequent vs. Popular words

As we can see in **Table 5**, candidates are using often positive words as "Great", "Thanks" and they also refer to events that happened or will happen, using "Today", "Tonight" etc. Only in case of Hillary Clinton and Rand Paul we can see some piece of information from their actual program - Hillary mentioning women rights and family and Rand Paul mentioning taxes. In case of popular words in Table 6 we can see more information about the supporters of the candidate.While Jeb Bush's and Ben Carson's followers appreciate religious topics, Rubio's followers apparently like his enthusiasm to women's national soccer team.

7. DISCUSSION

7.1 Topic Extraction

Problem of extracting topic from the text is the huge topic and it deserves much more space in this paper. As the time was very limited, we couldn't come up with solution which would be 100% sufficient. However, we tried multiple approaches and got results which we could use through this project for further analysis. Anyway, one fact is clear - In this kind of project, where result depends on understanding of topic, it's extraction is crucial element and rest of the project depends on it. Thus, in case there was more time for this project, this would mostly probably be the area where we would invest it.

7.2 Popular Topic Recommender System

Our "Popular Topic Recommender System" can surprisingly give many good suggestions, even with such a limited dataset, a select amount of tweets from few users. From the system, we discovered that SVD is powerful, it can classify topics without knowing the anything about the words' meaning.
Twitter accounts such as DEMdebate, GOPdebate and KeepShoutingOnGuns are useful for collaborative filtering, which is beyond our expectation. It may because those accounts are interested in some specific areas, hence they can link up related topics.

7.3 Limitation

In its curren state, the "Popular Topics Recommender System" carries several limitations that would require more work.
When doing topic extraction, the Natual Language Processing Toolkit (NLTK) cannot extract all the nouns from Twitter posts correctly. For example, in the following Twitter post:

In the past two Republican debates, no one said a word about equal pay. Maybe the third time's the charm? #GOPdebate

Since NLTK recognizes the part of speech word by word, it cannot consider with respect to a whole sentence. In the post, "debates" is a verb but NLTK classifies it into a plural form noun. Apart from the part of speech problem, NLTK also cannot extract an important topic "Republican" from the post. Both of the problems are affecting the recommender system's accuracy.

The recommender system engine, Crab [2], is memory based, so it can only process limited posts. With one month's posts, it normally takes a few minute to generate a recommendation on our AWS cloud server.
As the system extract topics from Twitter posts, obviously, it cannot make suggestions to the idea that doesn't appear in the posts before. Additionally, because of the limited number of posts, the system needs a user to input two to three ideas at a time for brainstorming suggestions.

7.4 Future Work

There are several areas where the "Popular Topic Recommender System" could be further developed for better performance.
For the quality of recommendations, apart from using the lemmas of nouns, more complex suggestions can be made by using verbs, adjectives and adverbs. Giving stronger multiple words recommendations, such as "beautiful day", "very sad moment" as well as applying advanced machine learning technologies like collocation extraction can also help.
Besides, we can introduce a depreciation factor which is in the range of [0,1] to the "Topic Popularity Rating". Such that older posts will have lower rating than the latest posts. With this formula, we can train the recommender system with more posts and keep the suggestions trendy at the same time.

$$R_{popularity_new} = R_{popularity} \times Depreciation$$

We could also collaborate Twitter with Facebook in order to get more content for training.

8. CONCLUSION

We found our project interesting and we enjoyed trying out different methods we learned during the course. We found out that working with big data is sometimes a lot of experimenting. Time for this project was quite limited and if we had more time, we could try more different approaches and tweaks to the methods we used. However, we think we made a good job and showed our ability to apply knowledge we learned in this course.
In this project we got our hands on many various techniques for processing and transforming data. We used with the real data produced by people on the internet, so we learned some valuable lessons during applying the methods and techniques we learned during the course.
When designing a solution for our project, we spent a lot of time considering different approaches and had to pick the most suitable one. Usually it was hard to tell if a selected approach would work, so we had to experiment. This led to some effort of trying to implement different methods, only to find out that the approach would not really work, meaning that we had to iterate upon those ideas and try something different.
We believe that by working on this project we improved our knowledge, as well as strengthening our sense of thinking, about data and picking the right method to achieve particular goal.

9. REFERENCES

[1] Natural language toolkit.
[2] I. Medeiros. Crab - recommender systems in python.
[3] NLTK. Collocations.
[4] B. Santorini. Part-of-speech tagging guidelines for the penn treebank project (3rd revision). 1990.
[5] Stanford. Stemming and lemmatization.
[6] M. Taylor. Twitter api wrapper in python.
[7] H. Wilson. a week is a long time in politics. mid-1960s.

APPENDIX

A. SOURCE CODE

A.1 Popular Topic Recommender System

Source code, cross validation data and readme are attached as popular_topic_recommender_system.zip.

A.2 Map Reduce

Source code, demo input and readme files are attached as MapReduce.zip.

A.3 K-means Topic Clustering

Source code, demo input and readme files are attached as kmeans_topic_clustering.zip.

A.4 Data Obtaining and Pre-processing

Source code are attached as data_obtaining_and_topic_extraction.zip.