Summarizing and Visualizing News Articles by Using NLP Techniques and Neural Networks

By
Vinay Kumar Reddy Chilakala
Sai Krishna Devanapally

Abstract:
In the current era of artificial intelligence, our everyday technologies such as SIRI, Cortana, etc. work on the basis of Natural Language Processing (NLP). Here we have concentrated on data generation which is gathered through a scrapper and processed with NLP to generate the recommendations for the user depending on his search criteria. Implementing Neural Networks in this schema to find the weight of the various results found and suggest the best one to the user makes it reliable for recommendations and searching.

Introduction:
Today, a huge amount of data is generated from various sources of data such as news articles, web pages, databases, etc. most of which is text. And in the modern era of smart machines, everything depends on how a machine can interpret the input given by the user and how well the machine responds to it. The process of machine understanding the input given by the user in the form of a human interpreted language can be done by using Natural language processing. This can also be used to interpret data from text documents written in any human language.

Various natural language processing tools are available in the market which starts with basic text analysis to complex sentiment analysis which is either freeware or paid versions. When it comes to Natural language processing libraries some of the best available in the market include Stanford core NLP Suite, Apache OpenNLP, General Architecture for Text Engineering(GATE). Of all these libraries Stanford Core NLP is the best to work with due to its high flexibility and a large number of features.

In our project, when a user searches for a word over web, we are suggesting a recommendation system which is based on neural networks to find the weight of different articles found over web and collects the articles and by using natural language processing techniques in the found article to find the frequent terms which occur along with the search word and suggesting articles with respect to that word.

Technologies:
JAVA 8
Stanford CoreNLP Suite
D3.js
Webhouse.io

Stanford CoreNLP Suite:

Stanford CoreNLP is an open source natural language processing library which is written in Java and was released by Stanford natural language processing group. They have released a lot of natural language processing tools which include CoreNLP, Parser, POS Tagger, Name Entity Recognizer, Word segmenter, English tokenizer, Relation extractor. Each of these has different specific functionality which can be used depending on the user requirements. Stanford NLP group actively keeps updating their libraries to extend the bound of natural language processing

In our project, we used CoreNLP which can process languages such as English, Spanish, Chinese, French, German, Arabic. This is one of the best open source libraries to work with different languages. CoreNLP is written in java but designed to work with JavaScript, Python and other languages. It includes tokenization, part-of-speech, parsing, named entity recognizer and coreference. We have made use of tokenizer which splits the sentences into word tokens, named entity recognizer to identify the named entities such as the name of a person, place, location. Parser to analyses the logical components of the text. Coreference to identify the term in the text with respect to named entity recognizer which also refers to the same named entity recognizer.

Figure 1. Example of CoreNLP. (Image credits: https://stanfordnlp.github.io/CoreNLP/index.html)

From the image above which states the named entity recognizer is used to identify the named entities such as person, place, date, time, etc. Coreference point outs the relation between the words which are used previously in the text. One good example of coreference is, it can be used to identify pronouns of the same name or pronouns with a different context.

D3.JS:

One of the best and most user-friendly libraries to work with data visualization. It is a JavaScript library. It is extremely fast and supports dynamic behaviors for interaction and animation. It can be used to generate HTML, SVG, and CSS for more flexible work environment. It can work with full capabilities on any modern browser. It uses Document Object Model(DOM) on the given data and applies data-driven transformations to generate the visualization.

Webhose.io:

One of the most reliable web scraper to work with. We have integrated WebHoseClient in our java source code to obtain the data over the web with respect to our search criteria. One and only major drawbacks of this software is, it is a paid version. In the free version, we can only send 1000 queries which is the given limit in a free edition. It can gather and save data in many formats such as XML, JSON, RSS depending on user requirements. One of the advantages of using this Client above others is we can code it to crawl and scrape data from different sources such as articles, news, comments, blogs, reviews and merge them into a single API.

Challenges faced:

The initial idea was to implement the Natural language processing techniques in spark environment but due to compatibility issues between spark environment and Stanford CoreNLP, we had to implement Natural language processing in the local environment.
The cause for the compatibility issues between them is due to the recent updates in Spark which removed few libraries which supported the linking of Stanford CoreNLP with spark environment.

Next challenge was to find how to give priority to a large number of articles which were given as a result of searching the keyword over the web, the solution was given by neural networks which were used to give weights to different word along with the keyword and keyword itself. The weight of the individual paragraph is weighed and the ones with the highest weight or the least weight depending on user requirement are given as output.

Process Flow:

The first step of the process is gathering data which is done by web scraping using Webhose. The data extracted is given as an input to Natural language processing libraries, where the process of finding named entity recognizer given by the user and using coreference functions of the CoreNLP to find the related words and the words which occur along with it.

From the data generated from CoreNLP, we can interpret the words which occur the most along with the keyword. Weight is allotted to each word in the article and it is used to give priority to the articles which occur along with the search word. Using this we can generate user recommendations to the user depending on his search criteria.

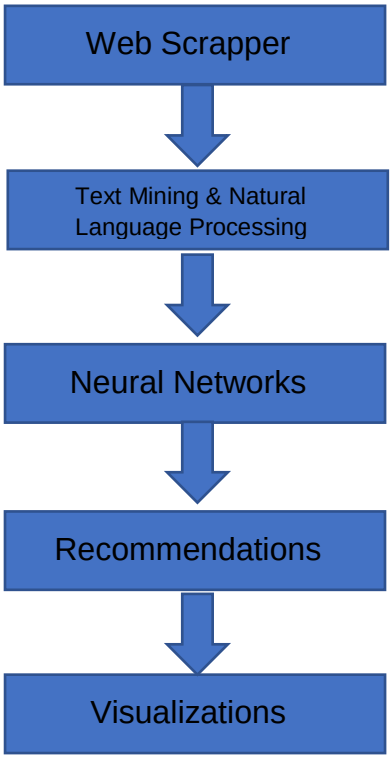

Figure 2, Process Flow

Methodology & Results

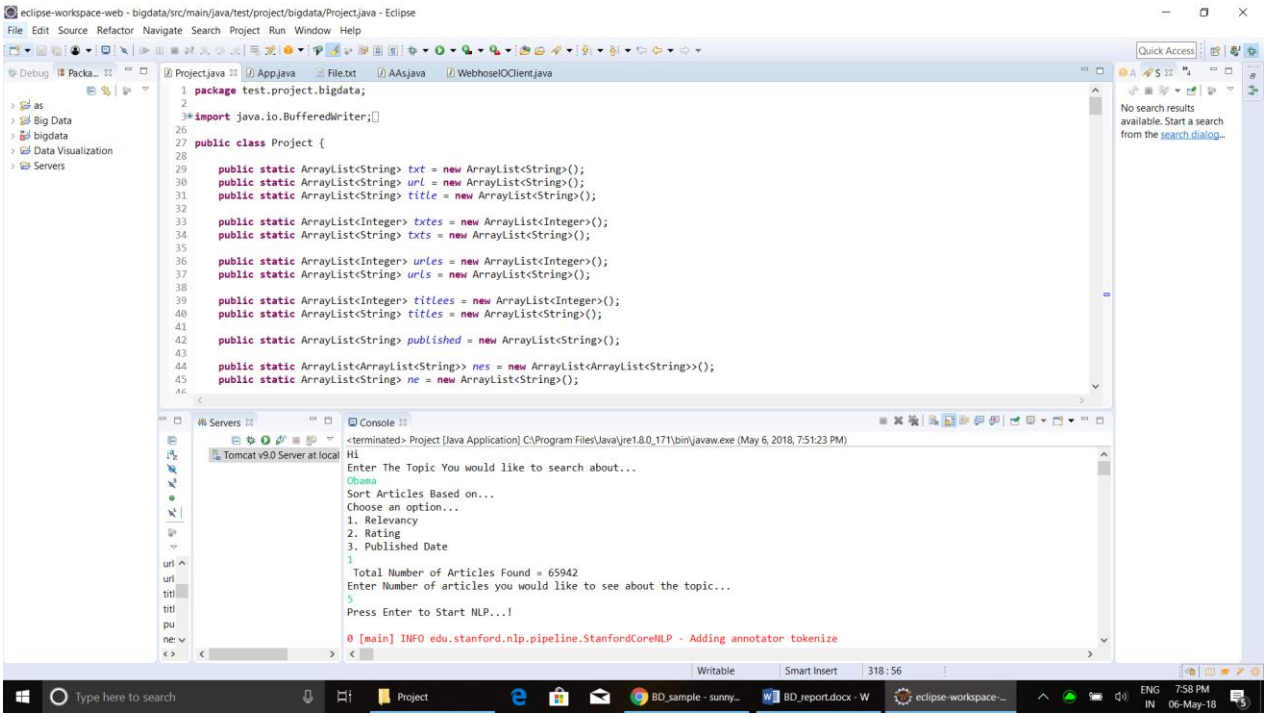

Figure 3, Program Interface.

As we run the code, the first thing we enter is the query word, Say "**Obama**".
Then we get 3 options to choose from, on what basis do we want to sort our results.

```
1. Relevancy
2. Rating
3. Published Date
```

Then we get the output. The code returns a number of articles found for the Query.

Out of all those articles, we can enter how many articles are we interested in reading rather than displaying all the hundreds or thousands of articles found.
As we fire up the NLP. The actual code comes Into Picture.
Now the Articles are pipelined for Natural Language Processing. NLP is going to analyze each and every word and give an NER tag. Each Named Entity is used by Neural Network to branch and then add weights. Each sentence will be processed to understand whether this sentence talks about the searched query or not. Based on the weights and the total number of sentences that talk about query gives the score for each article. This score is used in Recommendation Algorithm. Also Named Entities created by NLP is also used to Identify key terms related to the article. These key terms are again sent to Webhose Client to fetch articles related that keyword. Two articles are recommended after every main article about **Query**. These two articles are related to 2 important words found in the main article.

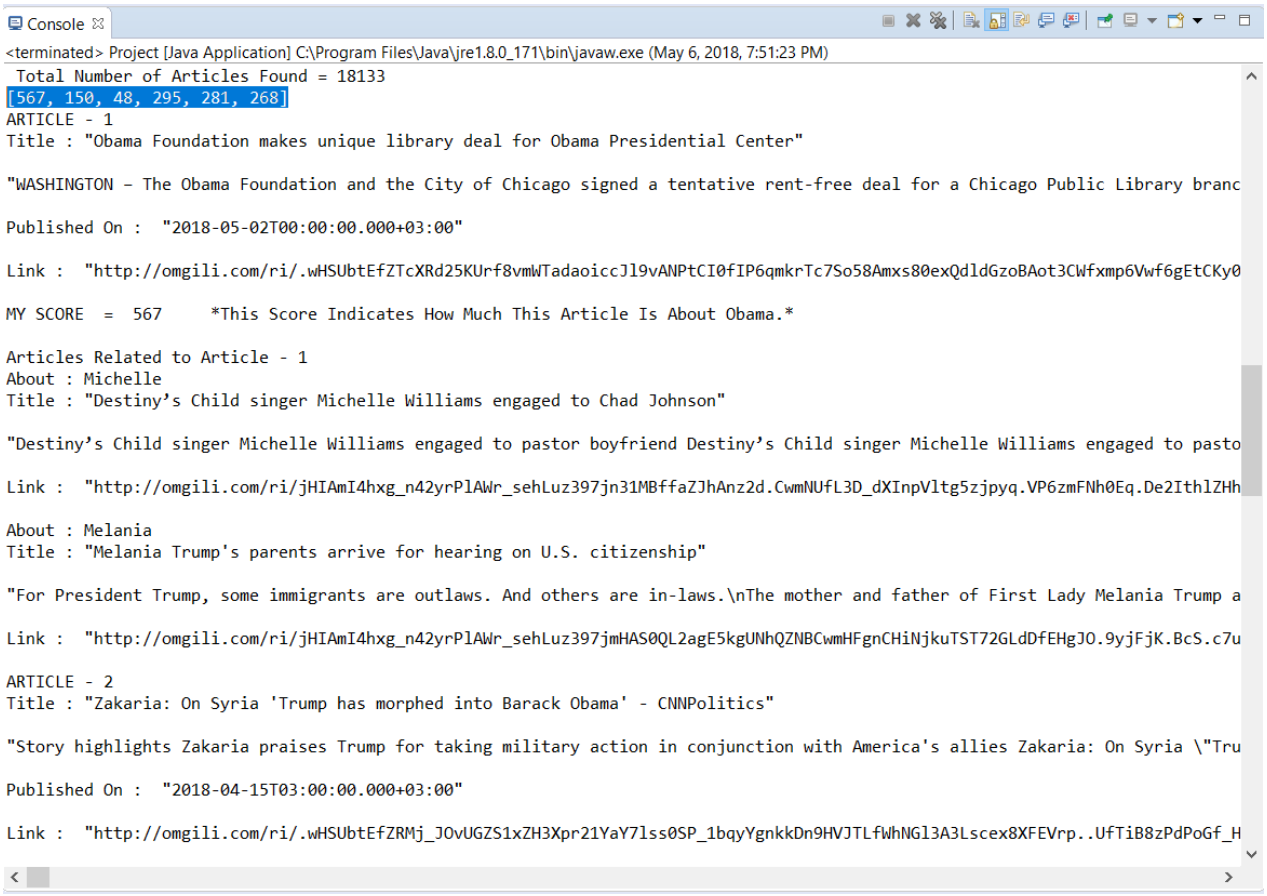

Figure 4, My Score Array for Main Articles 1-6.

Each Article is displayed with Article Number, Title of the Article, Entire Text in the Article, Published Date, Link to Open the Article, and finally Recommendation Algorithm Score for that Article. This Main Article is followed by Two other Articles Related to the Main Article.

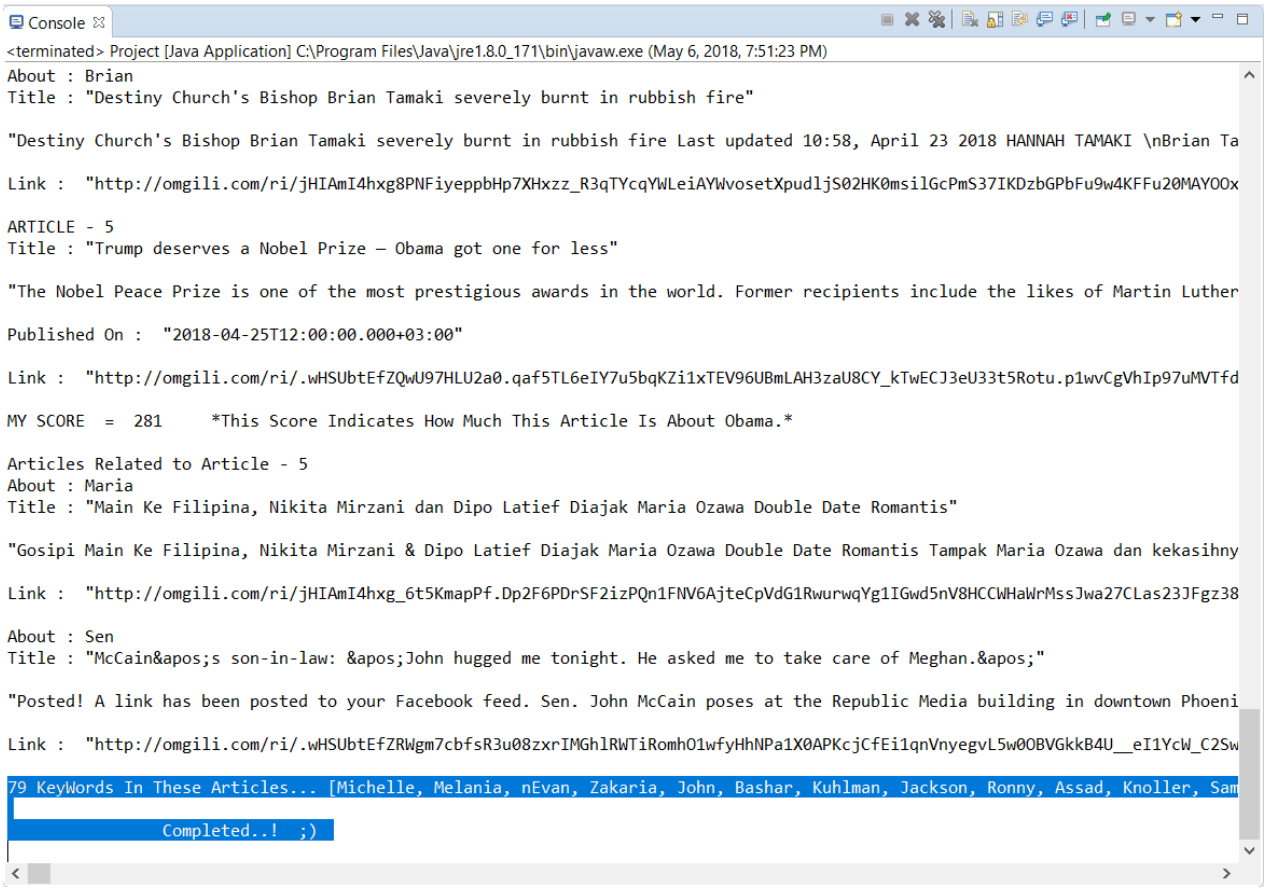

Figure 5, All the Keywords Identified in 5 Articles.

All the Named Entities identified in all the main articles are checked if they belonged to these 3 categories.

1. Person
2. City
3. Nationality

If so then those words are added to an Array and then displayed in the end. The user can find any of these keywords interesting and search for them next time.

Visualizations

A part of this project has been used in Data Visualization Project. Completely different code has been developed to do Web Scraping and then Natural Language Processing has been used to Identify what each word is and then have a Count on Number of times each Keyword is Repeated. This Visualization can also help in doing Recommendations.

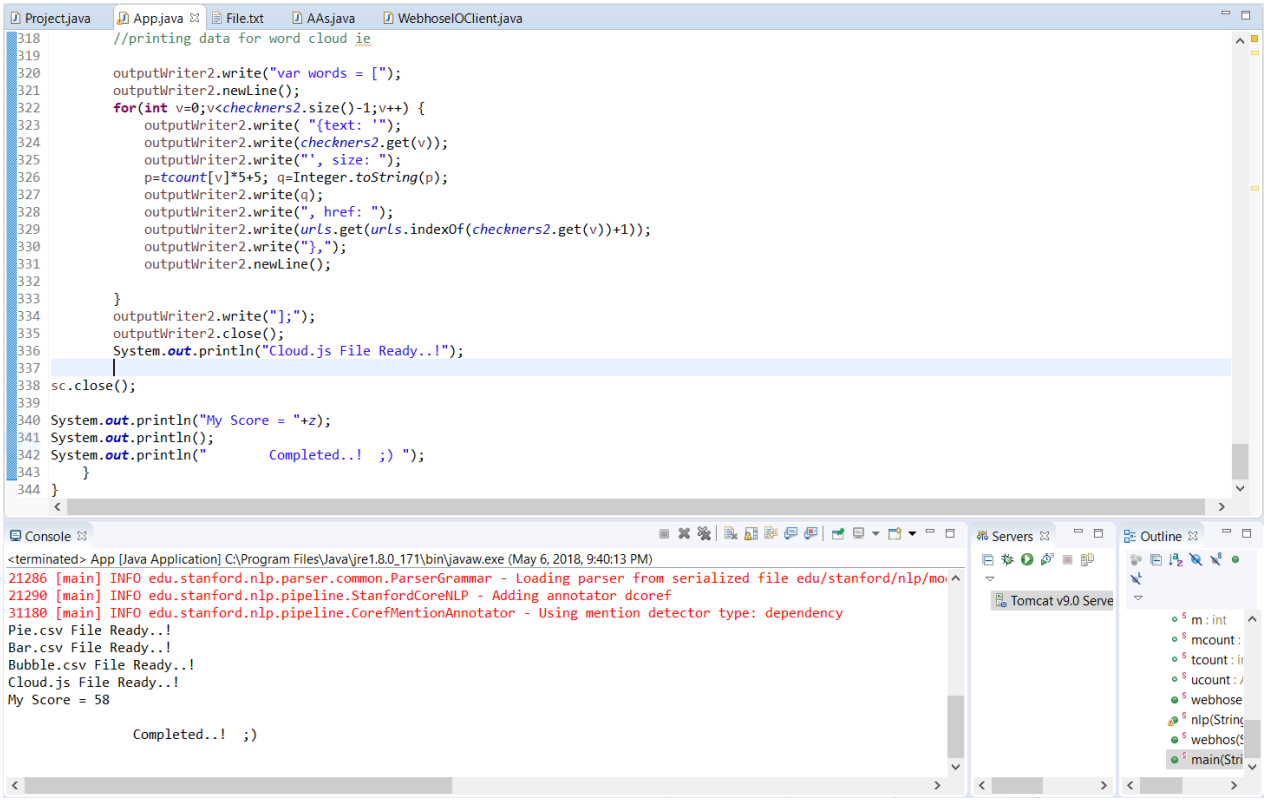

Figure 6, Visualization Program Result.

Bar Chart Visualization

This visualization shows how many URLs are available for each Named Entity Recognized When searched about Games. Clearly, Gamers and EAstports have more links for the searched article.

Sometimes other words might have more URLs than the article user actually searched about. Such interesting keywords can be identified easily which can suggest user to open next article.

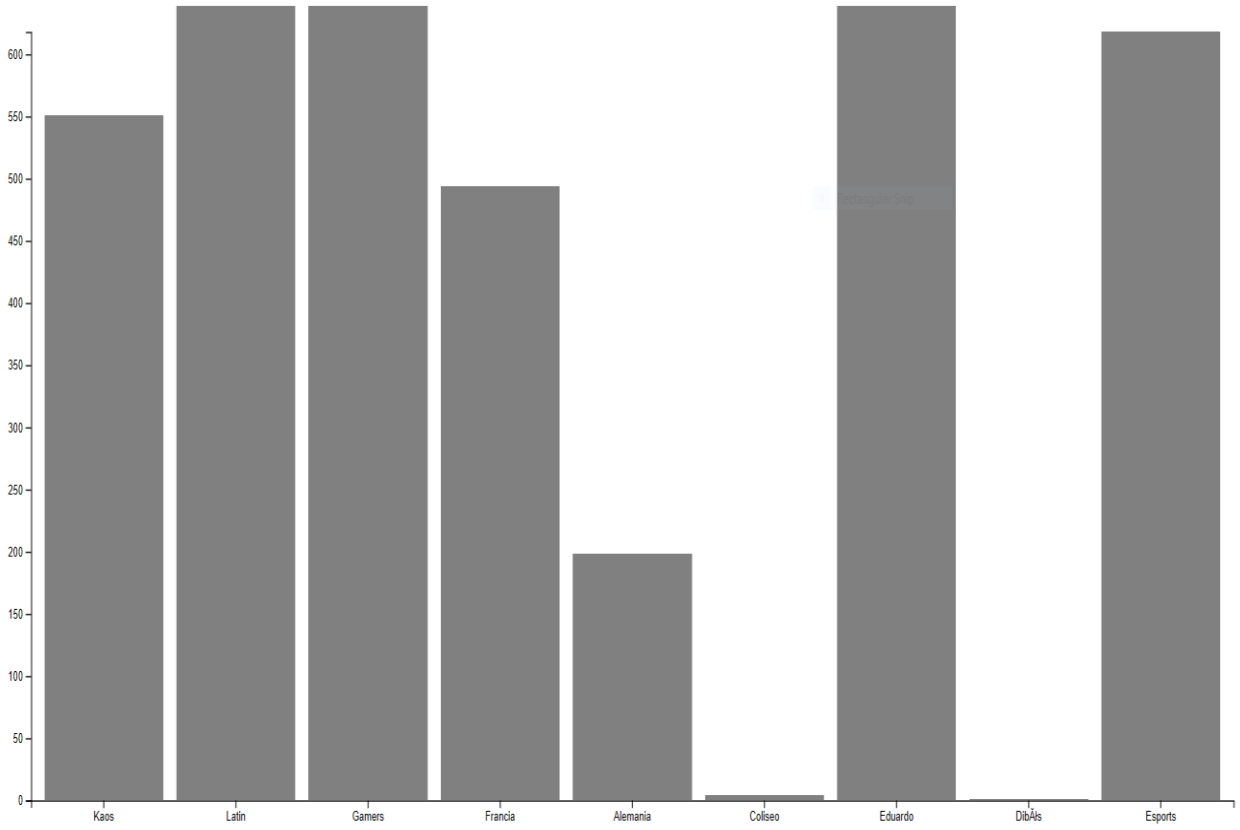

Figure 7, Bar Chart Visualization.

Pie Chart Visualization

Next Visualization displays how many times or what percent of the times each important word has repeated. This makes end user understand how many sentences of the article he has searched, speaks about these words.

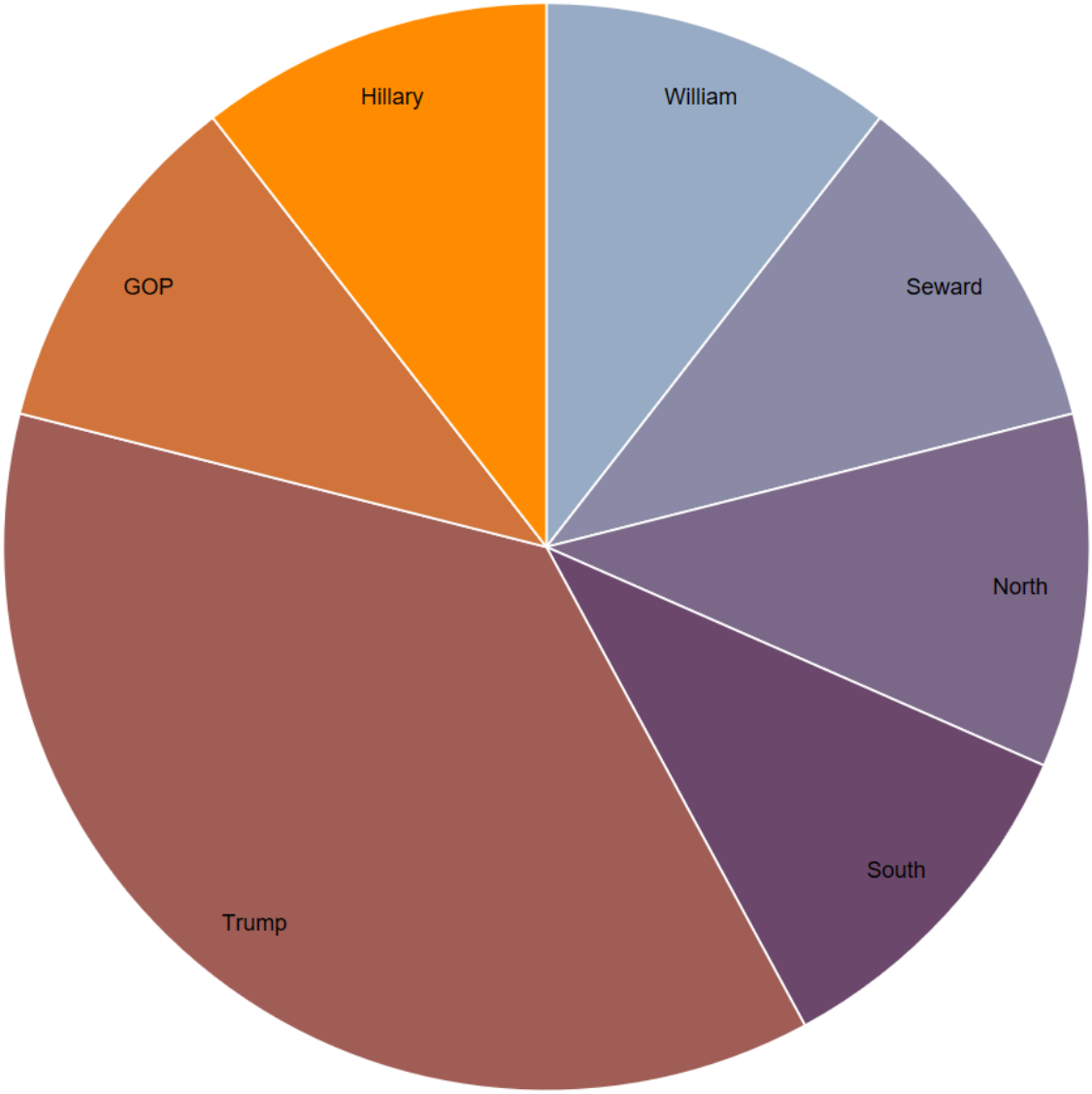

Figure 8, Pie Chart Visualization.

Bubble Chart Visualization

This visualization gives the count of number articles available for each keyword and pictorially the size of each bubble represents the number of articles with respect to each other. This visualization is intractable. It displays a number of articles available respectively when the mouse pointer is moved over the bubble.

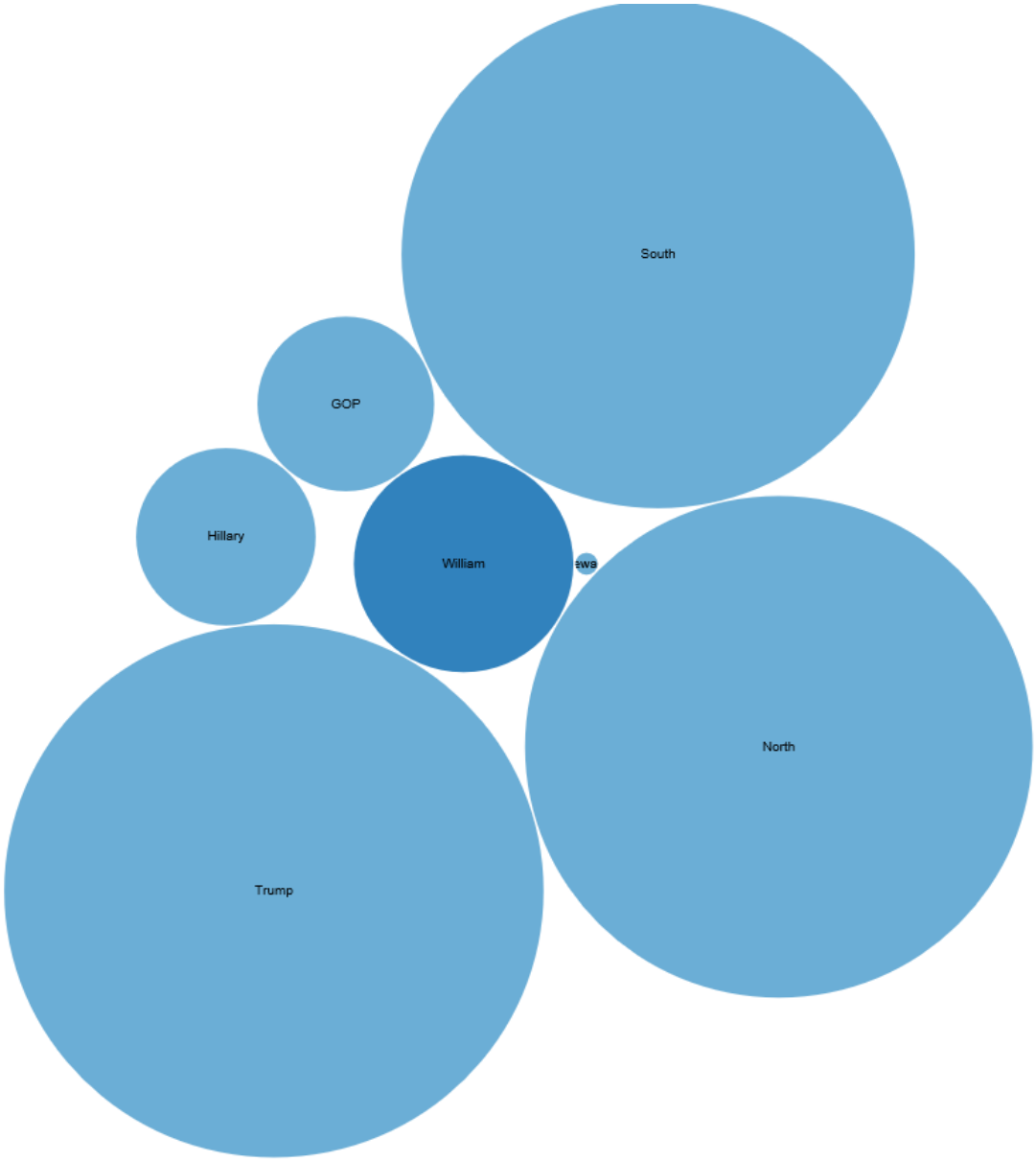

Figure 9, Bubble Chart Visualization

Word Cloud

This is final interactable visualization which has links in it to open respective articles. The user can get an idea about which article or keyword he should open based on the previous visualizations.

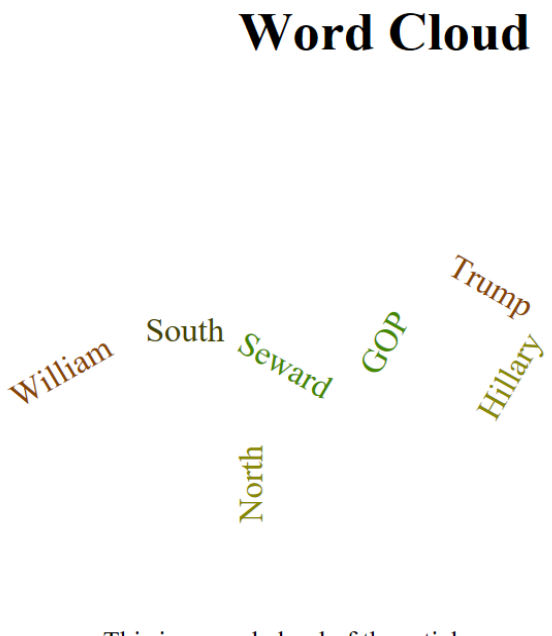

Figure 10, Word Cloud Visualization.

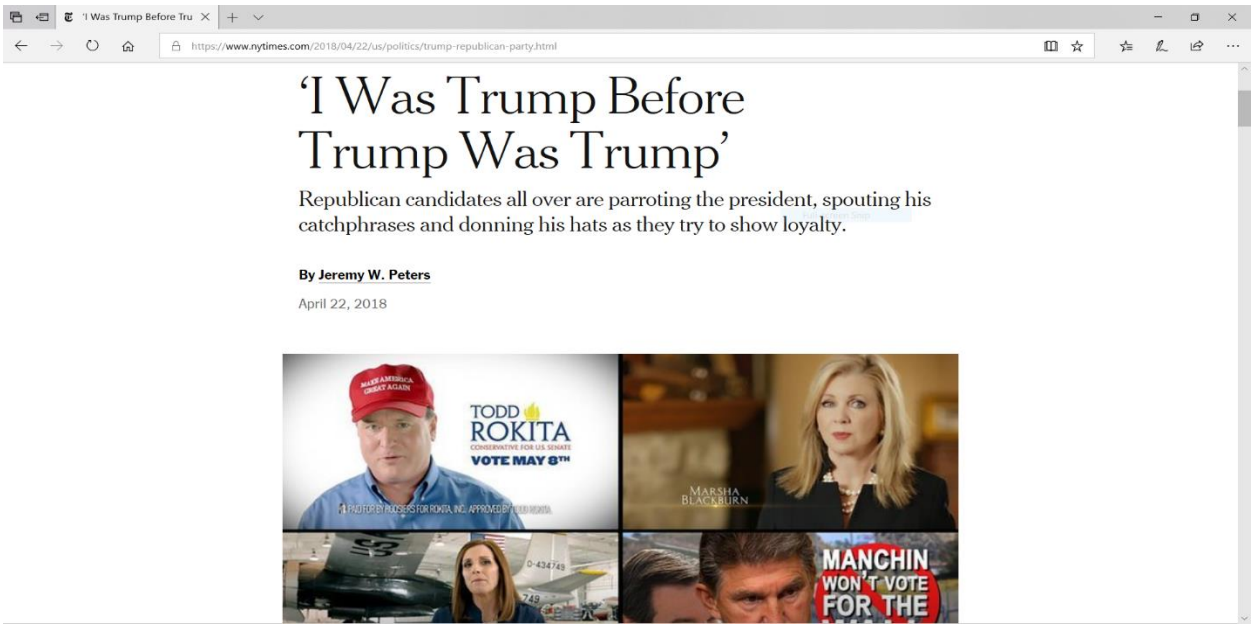

Figure 11, Word Cloud Link

Future Work

For faster and more optimized recommendations in current technologies where big data is a thing, we could implement Natural Language Techniques in spark environment to give faster recommendations. We can also implement phrase search in the spark environment using Natural Language Techniques to give find results for phrases given by the user in a human language like English.

Conclusion

This Project has given me an opportunity to work on Data Mining or Text Mining, Neural Networks, Recommendations, Predictions and many other Big Data Concepts in Real Time Scenario. This Program we developed does not have any References. This project is the realization of an idea.

Acknowledgment

We would like to thank Stanford Natural Language processing group for providing all the required documentation and helping with the errors during the implementation of NLP on the local machine.