Document Summarization Using NLP and Big Data Techniques

Abhirup Bhattacharya
Department of Computer Science
University of Texas at Dallas
abhirup.bhattacharya@utdallas.edu

Manasi Barhanpurkar
Department of Computer Science
University of Texas at Dallas
manasi.barhanpurkar@utdallas.edu

***Abstract*— A versatile method for summarization of data using NLP (Natural Language Processing) based techniques is proposed for dealing with memory management with verbose data and articles. TextRank a ranking algorithm which makes use of a graph-based approach to process text. In this paper we discuss the usage of TextRank in accordance with sentence similarity vectorizers for the summarization of news articles scraped from the internet.**

Keywords— NLP, TextRank, document summarization

I. INTRODUCTION

Today on average a single news website publishes at least a 1000 articles per week. Storing these articles and extracting meaningful information is a very important part of archiving information for future uses, thus it makes sense to store the summary of these articles to save pertinent information but also save memory while doing so. We have thus tried to marry NLP (Natural Language Processing) techniques with Big Data technologies to make this possible.

We are currently in the era of big data, collection and analysis of data in today's age is the foremost priority not only to map previous trends but also predict and model future capabilities. With ever expanding data it is important to expand memory reserves as well as be stringent with its use. It is also important to have enough computing power to extract meaningful information from this stored data, this is usually done through distributed means.

For the sake of extracting meaning from contiguous data, NLP techniques are the most efficient methods outlined. The process of extracting summary of text from data requires us to work with sentence similarities like TF-IDF, cosine similarity etc. to outline the most important words in the document as well as reduce redundancy.

II. DATA SUMMARIZATION

Text summarizing is a useful technique for extracting important information from text documents. Using NLP techniques, it generates a concise summary of the document. There are two approaches: extractive text summarizing and abstractive text summarizing.

A. Extractive text summarizing

This is a straightforward technique that doesn't aim for fluency but rather focuses on the importance of the words and does not take into consideration the context. It selects a subset of words that retain the important points. It is effective as it takes lesser computing time and retains original words as summaries from the text.

B. Abstractive text summarizing

Abstractive text summarizing is a close match to how we summarize text as humans. It takes into consideration the context of the text and makes a smaller more concise but different sentence with the same meaning. This requires a higher level of abstraction and also is more fluent and coherent than its counterpart.

III. TEXTRANK (GRAPH BASED RANKING ALGORITHM)

A graph-based ranking algorithm ranks nodes in a graph by taking global information derived recursively from the entire graph into account rather than just local vertex-specific information.

TextRank is built on the foundations of PageRank. PageRank is a ranking algorithm well known as the ranking used by Google, it was formulated by one of the co-creators of Google – Larry Page. PageRank is a method for determining the weight of a web page. Consider each web page to be a single, massive, directed graph. A website is represented by a node in this graph. A link can be thought of as a directed edge that runs from page A to page B. The fundamental concept is the same because the text in TextRank is the webpage in PageRank.

"Voting" or "recommendation" is the core idea behind a graph-based ranking system. A vertex effectively casts a vote for itself when it is connected to another vertex. The number of votes a vertex has received determines its weight. The weight of the vertex casting the vote as well as the weight of the actual vote are both taken into consideration by the ranking model. The vertex's final score is obtained in much the same way we obtain PageRank, it includes both the ingoing and outgoing directed weights to reach its final score controlled by a threshold.

A vertex's final score is thus ultimately dependent on the vote casted. Let E be a subset of V X V and let G = (V, E) be a directed graph with vertices V and edges E. Vertex score is calculated using:

$$S(V_i) = (1-d) + d * \sum_{j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

Here, d is a damping factor with a value ranging from 0 to 1, and whose purpose is to incorporate into the model the likelihood of hopping from one random vertex to another in the graph.

The algorithm starts by giving arbitrary scores to every sentence. This score is equal and same for all and does not in any way affect the working or final decision of the TextRank

scores, only the convergences in the vertex can make divergences in the final score. Then the process of casting the votes start, these votes affect the score and tell how important one sentence is when compared to another. Every sentence is compared with every other sentence, thus creating a very compute heavy recursive action. This is sometimes managed by using jaccard scores for faster traversing. The algorithm continues till it reaches a set threshold.

If in case the vertices extracted have imbalanced connections. i.e, if the number of connections between vertices is too high or two low, then we introduce weights to the graph. This weight is determined based on how strongly connected the vertices are, weights are relevant to the edges between vertices.

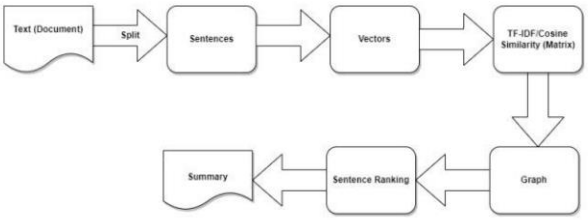

Fig 1: A step by step process of how text rank is implemented in this model.

IV. Summary Extraction

TextRank's primary application is the extraction of sentences for automated summarization. Keyword extraction, which deals with words or phrases, is similar to sentence extraction in concept. TextRank allows for recursively ranked text unit ratings based on full-text information.

This necessitates the construction of a network with nodes representing the rating units and links to the text. Because the goal of sentence extraction is to score full sentences, each sentence is represented by a vertex on the graph. To utilize graph-based rating algorithms on natural language texts, a graph that reflects the text and connects words or other text parts with permissible connections must be created. Because the text units are much larger, the co-occurrence relation used for keyword extraction is not used. Instead, we create a new link that evaluates the degree of content overlap between two sentences to determine if they are related based on their "similarity."

The connection between two phrases might be understood as a "recommendation" process. When one line in a text "recommends" that the reader read additional sentences in the text that address the same subject, a link can be created between any two such sentences with similar content.

The overlap of two sentences can be calculated by counting the common characters between their lexical representations.

$$Similarity(S_i, S_j) = \frac{|\{w_k | w_k \in S_i \& w_k \in S_j\}|}{log(|S_i|) + log(|S_j|)}$$

Where Si and Sj represent 2 sentences being compared. Wk represents the words at the particular iteration, It is imperative to note that we use the behavious of sentences from normal language. i.e, we think of sentences as an addition of words.

The cosine similarity/TF-IDF, longest common subsequence, and other similarity metrics can all be used to determine how similar two sentences are. The graph obtained is highly and densely connected, this graph contains weighted edges, the weight of the edge determines the strength of the relationship between the vertices obtained when pairing phrases. The ranking algorithm provides scores to every sentence. This score is used to extract sentences with high importance. The score is directly proportional to the score of the sentence and thus sentences are sorted in descending order of their scores and top n sentences are chosen as the summary.

V. Results

We make use of the dataset provided which is a set of news articles in relation to sportspersons in the field of tennis. We try summarizing the documents by giving every sentence a score using a similarity score as seen above. These sentences are then sorted based on this score and then the top scoring sentences are extracted as the summary. They all put together do not make sense, but they carry the main essence of the document. The cut off number is decided as a parameter and can be changed according to the "percent" of the document required as summary.

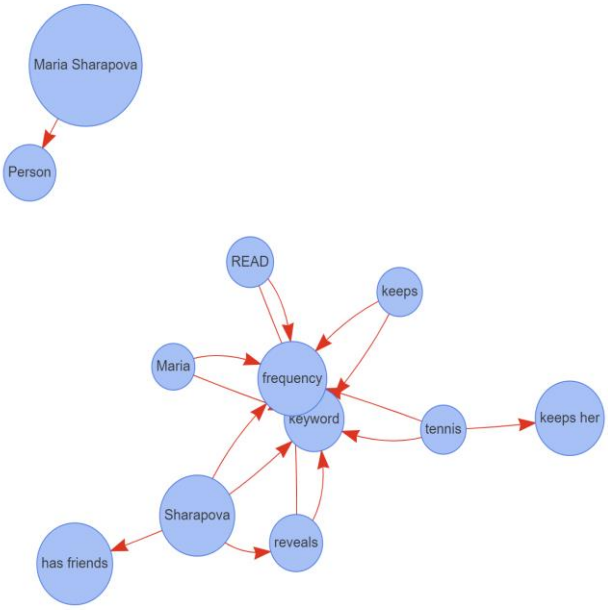

Fig 2[3]: A directed graph made using key information extraction from document1[4] of the dataset

Table

	Doducment title	Count of Sentence	Count of Summary
1	I do not have friends in tennis, says Maria Sharapova	19	4
2	Federer defeats Medvedev to advance to 14th Swiss Indoors final	12	3
3	Tennis: Roger Federer ignored deadline set by 'new' Davis Cup	15	3
4	Nishikori to face off against Anderson in Vienna final	13	3
5	Roger Federer has made this huge change to tennis - Lindsay Davenport	17	4
6	Rafael Nadal: World No 1 ARRIVES for Paris Masters - WATCH	12	3
7	TENNIS.COM PODCAST: POINT DEFENSE, RANKING DROPS AND ON-COURT COACHING	10	2

Fig 3: Table showing number of lines in the actual document vs number of lines in the summary generated using TextRank

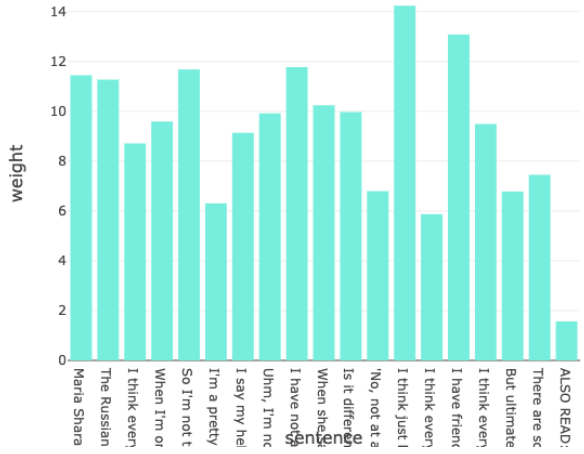

Fig 4: Graph showing score allocated to each sentence in document1[4] of the dataset

From the above figures we see that our model is successful in giving relative scores to every sentence and thus summarizing a nineteen line document into a 4 line summary.

VI. Conclusion

The Model for document summarization makes accurate and to the point summaries using the Extractive text summarizing technique. It makes use of TextRank to extract the sentences. In comparison to other models, the TextRank model works well to give accurate summaries without sacrificing context and meaning. This works well with statistical data, as seen from the dataset, statistics were preserved for almost every document that contained it.

The next steps for this project would be to attempt abstractive text summarization to make coherent sentence patterns that make sense in its entirety instead of sentences making sense only on their own.

References

[1] TextRank: Bringing Order into Texts Rada Mihalcea and Paul Tarau Department of Computer Science University of North Texas

[2] https://towardsdatascience.com/textrank-for-keyword-extraction-by-python-c0bae21bcec0

[3] https://github.com/BrambleXu/news-graph

[4] https://github.com/liuhuanyong/TextGrapher

[5] https://www.tennisworldusa.org/tennis/news/Maria_Sharapova/62220/i-do-not-have-friends-in-tennis-says-maria-sharapova/

[6] A Hybrid Model for Summarizing Text Documents Using Text Rank Algorithm and Term Frequency, Raghavendra Vijay Bhasker Vangara , Shiva Prasad Vangara , Kailashnathan Thirupathur , Bharath Tavidaboina International Journal of Advanced Science and Technology Vol. 29, No.4, (2020), pp. 3967 – 3975