CIS 511 - Introduction to NLP (Fall 2022)

Keyword Extraction and Text Summarization

Collaborators: Chanchal Agarwal , Faiyaz Ahmad, Grover Susanibar, Meeshawn Marathe and Vishal Patil

Abstract

Text summarization is the process of extracting important details from a text document. It is used to produce a succinct summary of the document, and this can be done in one of two ways depending on output: either by *extraction* or *abstraction*. One of the key methods in the field of text summarization is keyword extraction. The extractive approaches concentrate on choosing a group of words or phrases from the source text to produce the summary. In this work, we implemented four approaches - *Standard TF-IDF based sentence ranking*, *Luhn's* algorithm, and *Centroid based summarization* to extract keywords automatically and then summarize the text from an article from multiple newspaper articles. The fourth approach is the *TextRank* algorithm which is a graph-based summarization based on the *PageRank* algorithm. Finally, we performed a comparative analysis of all the methods. *Centroid based summarization* performs better compared to the other three algorithms.

Introduction

The amount of data available on the internet nowadays is increasing exponentially. With the increasing data comes the cost of time to process this data. Numerous free online newspapers are available to readers in the digital age. Every day, these newspapers include a wealth of information, making it difficult for readers to filter through all of it for the information they need. Here, an automated system that can only retrieve pertinent data from these news sources is required. In order to accomplish this, we need to perform text mining on newspaper articles. Text mining is the process of exploring and analyzing large amounts of data to obtain critical information. To execute the text analysis, text mining uses some Natural Language Processing (NLP) techniques including part-of-speech (POS) tagging, N-grams, tokenization, etc. It entails activities like text summarizing and automatic keyword extraction.

As part of this project, for text summarization of the news articles we have implemented four algorithms:

- *Centroid based summarization,* a text summarization method which extracts keywords and calculates the TF-IDF values for each. Further it summarizes the input text by comparing the cosine similarity between centroid vector and sentence vector formed using the TF-IDF scores.
- *Standard TF-IDF sentence ranking:* This method computes the TF-IDF scores for all the words in a given document and then computes the score of each sentence in the article using the corresponding TF-IDF scores for all its words.
- Luhn's method: Similar to *Standard TF-IDF sentence ranking*
- PageRank: A graph-based text summarization technique.

Related work

To establish foundation on current work in the field of NLP, we reviewed relevant literature to explore different approaches for keyword extraction and text summarization such as TF_IDF, Graph theoretic approach, and different ML approaches.

N. Moratanch [2] explains the use and importance of text summarization and differentiates between the two main summarization techniques, extractive and abstractive summarization. They described the way of experimental evaluation using a ROUGE-N method.
Pratibha Devi Hosur et al [3] suggest using unsupervised learning while performing automatic text summarization. This work presents a comprehensive overview of text summarization using NLP, including the text input document, preprocessing, the Lesk algorithm, and finally the generation of the summary.
Deepali K Gaikwad et al [4] This essay explains how the relevant details are taken from a lengthy text source and used to create a summary. Also, describes various methods for extractive summarization and abstractive summarization.
Samrat Babar[5] explains various extractive summarization methods and three main steps for summarizing documents - topic identification, interpretation and summary generation. Further, they also explained the two steps involved in extractive text summarization - preprocessing and processing. G. Vijay Kumar[6] mentions a model for text summarization using text rank algorithm for extractive summarization on unsupervised data.

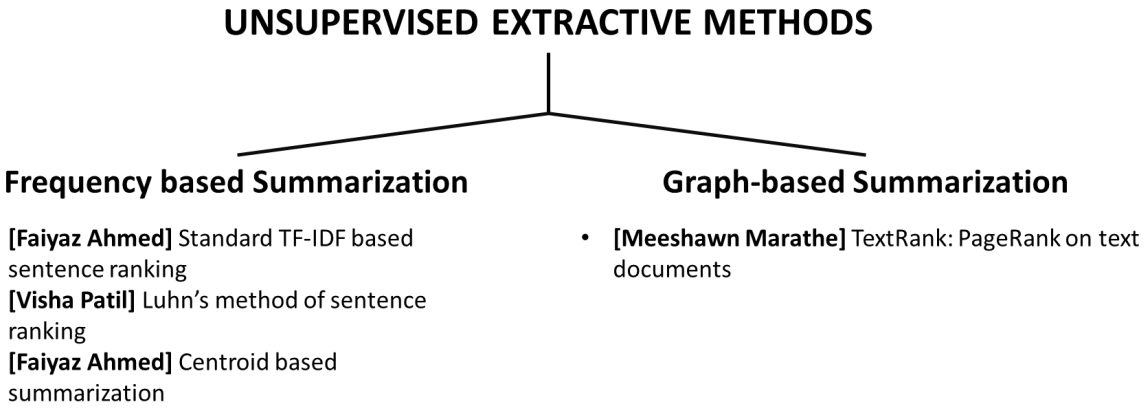
Dataset Description

We used the CNN News articles dataset[1] from Kaggle for the project. The English-language CNN/Daily Mail Dataset is made up of just over 3 million unique news stories that were authored by reporters for CNN and the Daily Mail. This dataset supports both extractive and abstractive summarization. The three data columns for each instance are id, article, and highlights. For each instance, there is a string for the article, a string for the highlights, and a string for the id. The news item in the "article" column is summarized in the "highlights" column. We will just use id and article because we are undertaking unsupervised learning for text summarizing.

	id	article	highlights
0	0001d1afc246a7964130f43ae940af6bc6c57f01	By . Associated Press . PUBLISHED: . 14:11 EST...	Bishop John Folda, of North Dakota, is taking ...
1	0002095e55fcbd3a2f366d9bf92a95433dc305ef	(CNN) -- Ralph Mata was an internal affairs li...	Criminal complaint: Cop used his role to help ...
2	00027e965c8264c35cc1bc55556db388da82b07f	A drunk driver who killed a young woman in a h...	Craig Eccleston-Todd, 27, had drunk at least t...
3	0002c17436637c4fe1837c935c04de47adb18e9a	(CNN) -- With a breezy sweep of his pen Presid...	Nina dos Santos says Europe must be ready to a...
4	0003ad6ef0c37534f80b55b4235108024b407f0b	Fleetwood are the only team still to have a 10...	Fleetwood top of League One after 2-0 win at S...

Methodology

Most of the current automated text summarization systems use extraction methods to produce a summary. Sentence extraction techniques are commonly used to produce extraction summaries. Broadly, the methodology can be explained with flowchart below. In this work, we use 3 TF-IDF based computation techniques for keyword extraction followed by text summarization and then a Graph-based sentence ranking approach.



Frequency Based Summarization:

1. TF-IDF Based Summarizer:

Term Frequency: For every word in given sentence we have calculated term frequency

$$TF = \frac{Number\ of\ Repetition\ of\ W_i\ in\ sentence}{Number\ of\ words\ in\ sentence}$$

Inverse Document Frequency: For every word in complete text given IDF Score.

$$IDF = log(\frac{Total\ Number\ of\ Sentences}{Number\ of\ sentences\ containg\ word\ W_i})$$

TF-IDF: Assigning weight for each word based on TF and IDF score

$$Weight(W_i) = TF * IDF$$

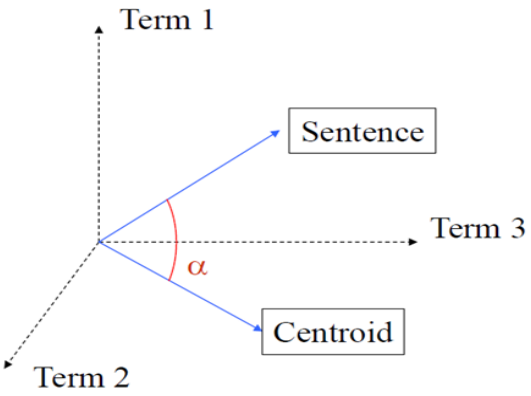
Sentence Weightage:

$$Sentence\ Weight = \frac{Sum\ of\ weight\ of\ all\ words\ in\ sentence}{Number\ of\ words\ in\ Sentence}$$

Sentence Extraction:

sentence weight> average sentence weight * tuning factor

2. Centroid Based Summarizer:



Centroid Vector: We have taken those words in a text which have higher tf-idf score than the average tf-idf score.

Sentence Vector: All the words along with tf-idf score in a sentence act as vector

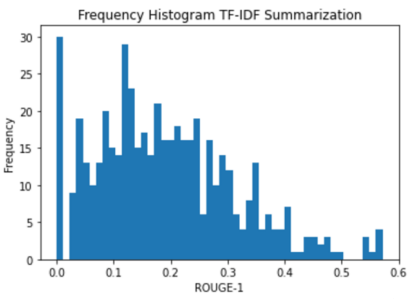
Cosine Sentence Similarity: Weight of sentence given based on the cosine similarity between the centroid vector and sentence vector

$$\text{cosine similarity} = S_C(A,B) := \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

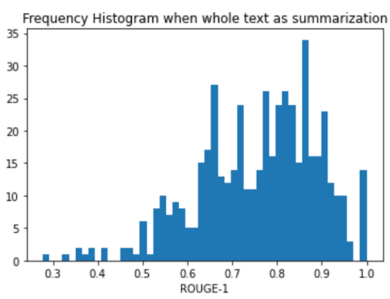
Sentence Extraction:

Cosine sentence similarity>avg cosine sentence similarity*tuning factor

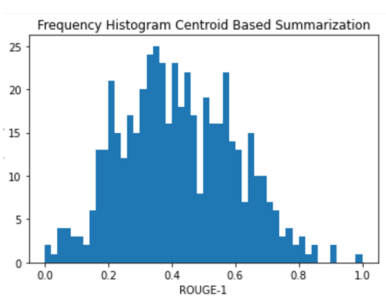
2.1 TF-IDF vs Centroid based summarizer



TF-IDF Summarizer
Avg. ROUGE: 0.18



Whole text as a Summary
Avg. ROUGE: 0.76



Centroid Summarizer
Avg. ROUGE: 0.42

All the above results are based on 500 instances of data.

Given that the gold reference summary is abstractive in nature, in this scenario, by using extractive summarization, the maximum ROUGE-1 score attainable will always be less than the ROUGE-1 score of the whole text as summary.

We are getting an average ROUGE-1 score as 0.18 for TF-IDF summarizer and 0.42 as Centroid based summarizer. It is evident that centroid based summarizers are performing better for giving the central idea of text.

3. Luhn's Algorithm Based Summarizer:

After cleaning the dataset i.e., removing punctuations and stopwords, lowering the case of the words and lemmatizing them, we calculate the Term Frequencies (TF) of each word in the sentences.

Then we calculate the Inverse Document Frequencies (IDF) of each word to determine the keywords. We select the top 10 words having the highest IDF score to get the keywords.

Next, in the Luhn's Algorithm function, we calculate the weight/score of the word (TF-IDF score) which is obtained by product of the TF and IDF score. The words having high scores are considered as frequent and important and the sentences containing these words are considered as important.

Next, we observed that the number of sentences in each news article varies and if the number of sentences in summary generated is very low, it does not give a clear idea about the context.

To overcome the issue of varying number of sentences in each news article, we computed a factor that generates a summary using 40% of the total number of sentences in the original news article. This improved the performance of the algorithm significantly.

In the end, we computed the evaluation metrics for the summary generated by the system by implementing the ROUGE-1 score.

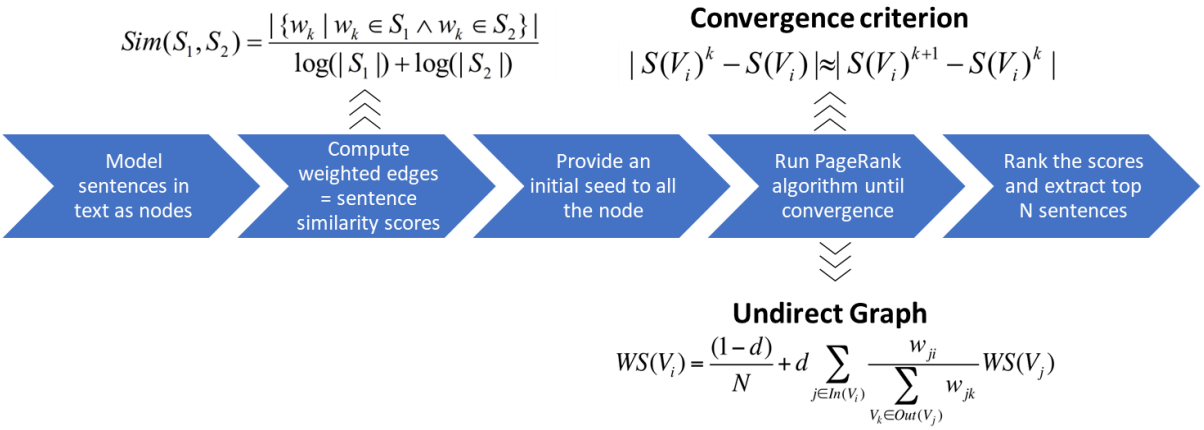
The performance of this approach could be improved by tuning the factor for the number of sentences in the summary generated.

4. PageRank Algorithm:

Brief overview:

- Its a *Graph-Based* summarization technique. Graphs can be directed or undirected.
- Extractive, and not abstractive.
- When applied on text documents, its called TextRank algorithm.

Computation workflow:



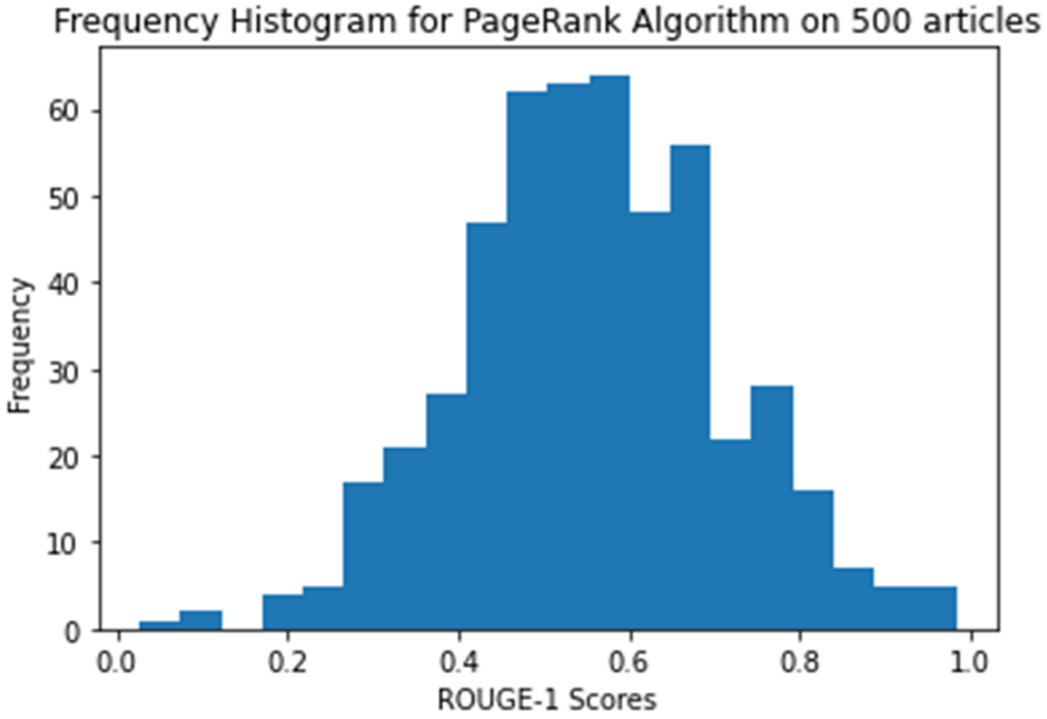
NOTE:

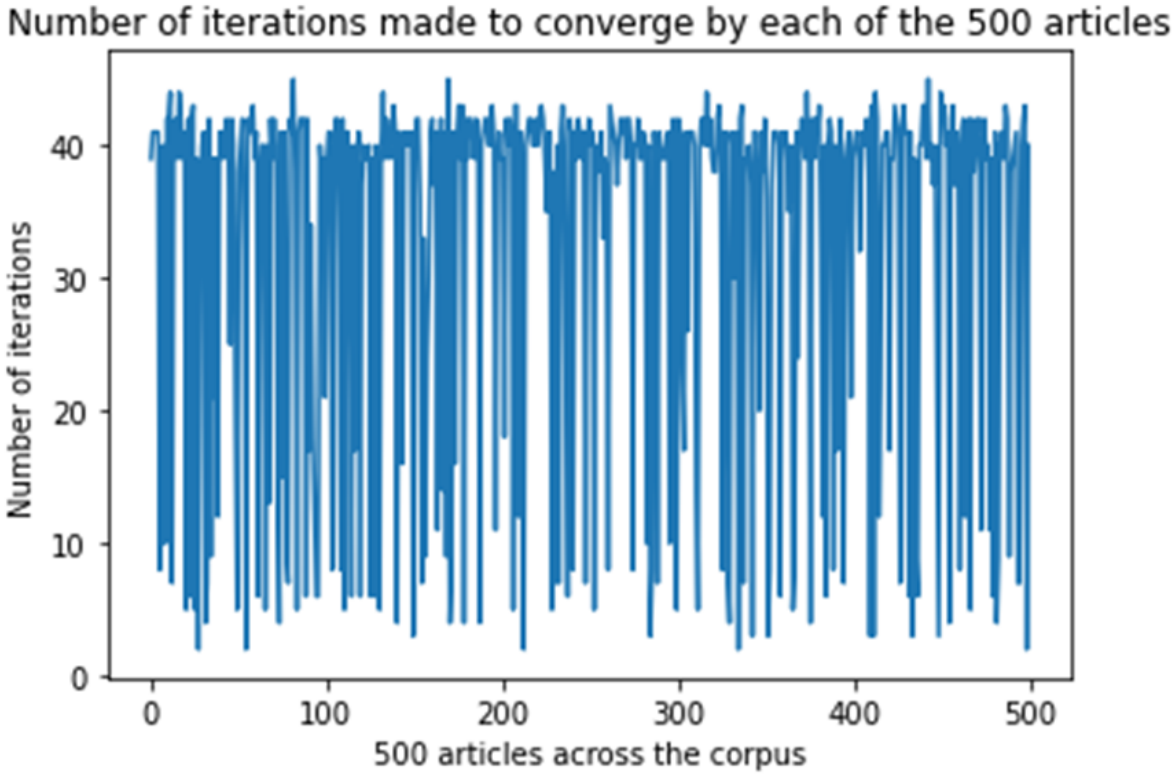
- For the TextRank algorithm, each sentence in an article is modeled as node in an *undirected* graph.
- The *Sentence Similarity* score quantifies the overlap between two sentences and is thus used as weights for edges connecting these sentences in a graph.
- Starting with an initial value, the weights of all the nodes are updated after every iteration of the PageRank score computation.

Implementation:

- The algorithm has been implemented in Python as a class with necessary member method to perform data parsing and cleaning and other functions that assist in the computation of the scores for all the nodes in the graph.
- The algorithm also computes ROUGE-1 scores to compare the model generated summaries with the annotated summary labels and prints a histogram of the score distribution across all the 500 articles.
- Tuning Parameters:
    - Initial seed value = 0.25 for all the nodes
    - Damping Factor = 0.85
    - Convergence threshold = 0.0001
    - N in Top 'N' sentences is modeled using a shrink_factor:
        - N = shrink_factor*num_of_sentences
        - shrink_factor = 0.15

Results:


Frequency Histogram for PageRank Algorithm on 500 articles
Frequency
ROUGE-1 Scores
0
10
20
30
40
50
60
0.0
0.2
0.4
0.6
0.8
1.0


Number of iterations made to converge by each of the 500 articles
Number of iterations
500 articles across the corpus
0
10
20
30
40
0
100
200
300
400
500

5. Extra NLP concepts

5.1 Topic Modelling

In order to figure out the mixture of topics from the research articles, a Latent Dirichlet Allocation model is trained over the corpus. The idea is to understand the latent probability distribution that generates the documents. Two results were obtained: the per-document topic distribution and the per-topic word distribution. The LDA model was trained with 25 iterations to get 10 topics. In Figure 3, we can see the 5 topics obtained and the per-topic word distribution.

```
Topic 0, title: people year, words: people,year,would,state,cnn,new,one,also,government,last
Topic 1, title: game player, words: game,player,team,year,club,league,goal,time,world,win
Topic 2, title: year one, words: year,one,mr,child,time,old,two,told,family,day
Topic 3, title: court judge, words: court,judge,police,case,murder,charge,prison,trial,victim,prosecutor
Topic 4, title: one water, words: one,water,new,also,food,world,per,could,people,company
```

Figure 3: Topic Modelling results

We can also get the topic distribution of each article. For example, in Figure 4, we can see that the first article (document 0) is composed by 39% of topic 4 (virus rna), 36% of topic 2 (protein cell) and 15% of topic 9 (respiratory infection).

	Topic 0	Topic 1	Topic 2	Topic 3	Topic 4
Doc 0	74.54	0.17	0.17	0.17	24.94
Doc 1	50.06	9.96	3.48	35.13	1.37
Doc 2	2.07	0.04	96.73	1.11	0.04
Doc 3	86.30	8.48	5.09	0.07	0.06
Doc 4	0.06	98.24	1.59	0.05	0.05

Figure 4: Topic distribution (%) of documents

5.2 Text Clustering

• PCA

Based on the article content, the idea is to find underlying features and group the articles that are in some way like others. In this case the input features would be the tf-idf values from the word dictionary. However, as there is a large dictionary (over 100’000 words), the clustering algorithm becomes difficult to train. Thus, first a Principal Component Analysis algorithm is run to reduce the dimensionality of the input features and find the components with more variance explained. As a result, 1’000 components are obtained per each document, and these are used as input features for the clustering algorithm.

• KMEANS

KMEANS is a clustering algorithm that at the beginning, it randomly assigns k cluster centroids to the input data points. Then it iterates until converge, and at each iteration for a data point it assigns the cluster with the closest centroid and then for each new cluster it computes the cluster centroid. Then the KMEANS clustering algorithm is executed, and 10 clusters were obtained. In Figure 5, we can see a graph in which each point represents a research article, and the color represents the cluster associated to each article. For this graph, the first two components of the PCA wereused as those components have the most variance explained from the input features.

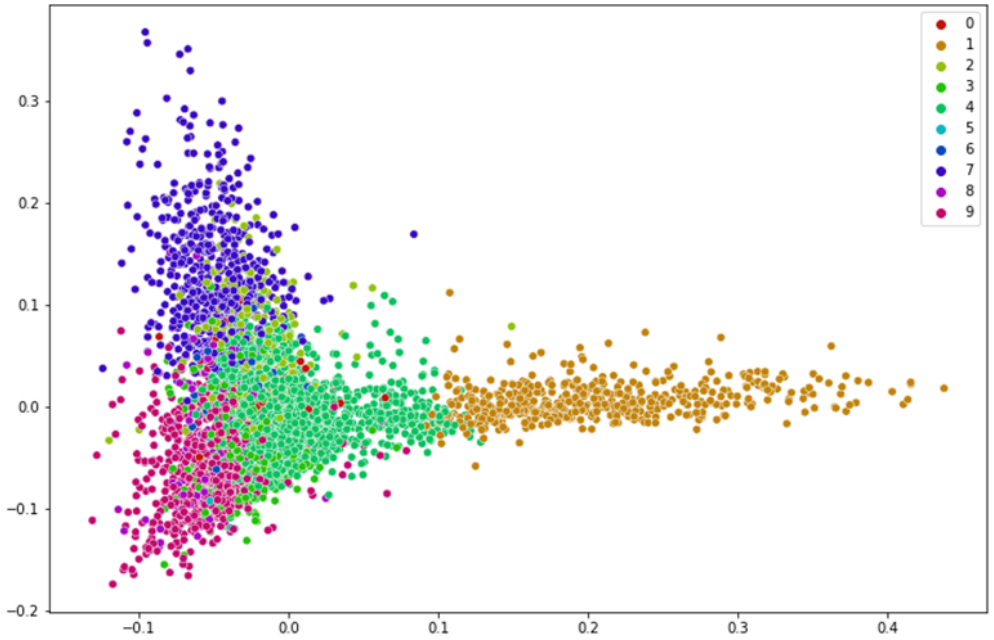


Figure 5: Text Clustering

For each cluster is possible to get the most frequent words in order to have an idea of the composition of each cluster. In Figure 6, we can see the Word Cloud of cluster 0, and we can say that the articles from this cluster talk about tech companies like Apple, Facebook and the interaction between users and devices.

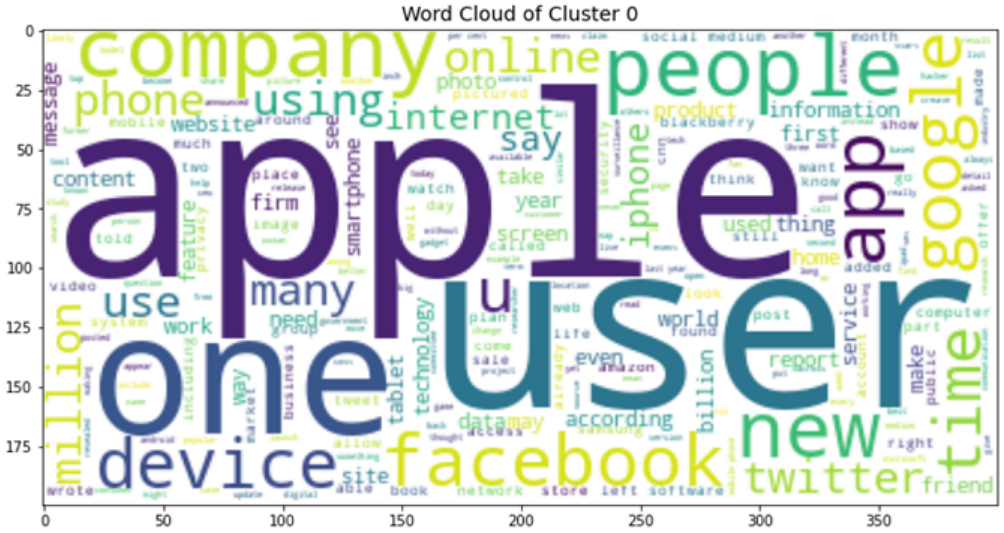


Figure 6: Word Cloud of Cluster 0

In Figure 7, we can see the Word Cloud of cluster 1, and we can say that the articles from this cluster talk about games, players, team, win, match, season.

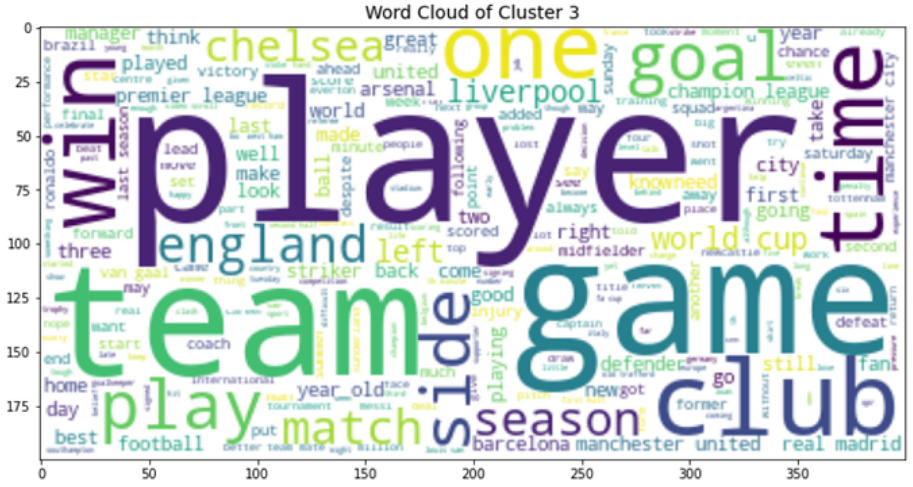


Figure 7: Word Cloud of Cluster 1

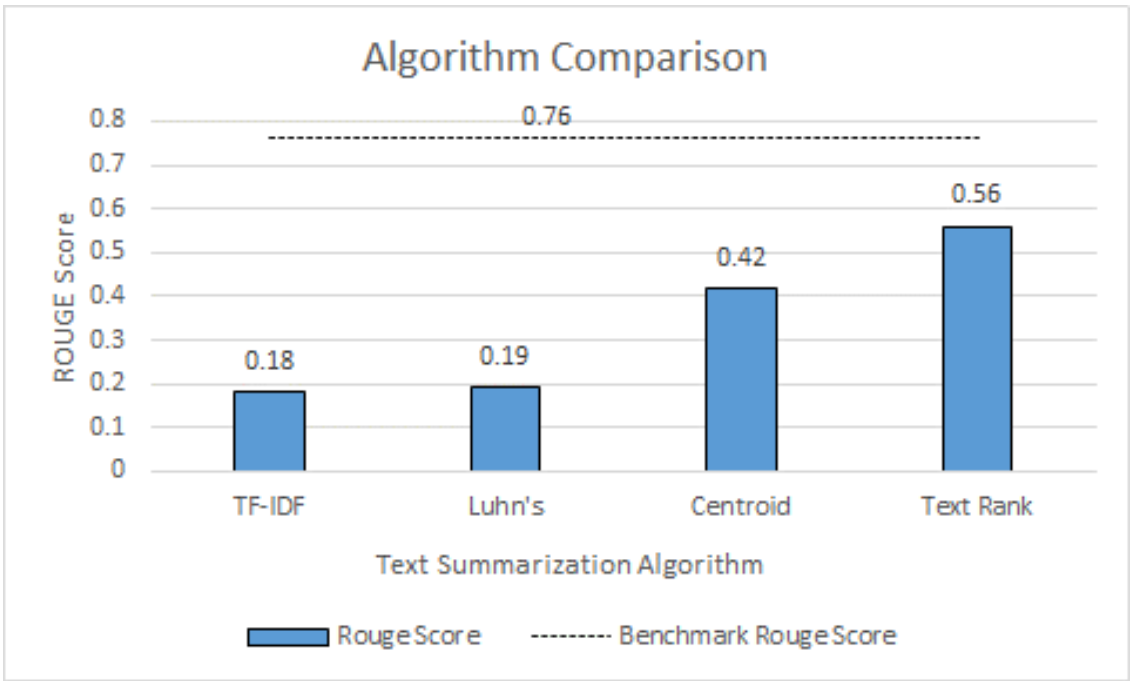
In Figure 8, we can see the Word Cloud of cluster 7, and we can say that the articles from this cluster talk about the government, Obama president, country, american people.

Figure 8: Word Cloud of Cluster 7

Experimental Discussion

To evaluate the performance, we employed the Rouge-N score. ROUGE toolkit, which is frequently used by Document Understanding Conference, is used to evaluate summarization performance (DUC). ROUGE-N, ROUGE-L, ROUGE-W, and ROUGESU are a few ROUGE methods. The fluency of summaries is estimated by the higher order ROUGE-N score (N>1).

$$\text{ROUGE-N} = \frac{\sum_{S \in \text{ref}} \sum_{\text{gram}_n \in S} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S \in \text{ref}} \sum_{\text{gram}_n \in S} \text{Count}(\text{gram}_n)}$$



Individual Contributions:

Chanchal Agrawal:

- Performed the literature survey and data collection for the project.
- Structured and designed the project.
- Performed the comparative analysis of different approaches.
- Worked on the data preprocessing and documentation of the project.

Faiyaz Ahmad

- Developed source code for the *TF-IDF and Centroid* based Algorithm:
    - Tokenizing the text into sentences.
    - Word tokenization for each sentence tokens.
    - Removing the stop words
    - Removing the regular expression
    - Computed the sentence weight for all the sentences
- Computed Evaluation metrics for the system generated summarization by implementing ROUGE-1 score
- Contributed to the making of presentation and report for TF-IDF and Centroid based Summarizer.

Grover Susanibar:

- Performed Topic Modelling
- Ran CountVectorizer, TF-IDF
- Performed Text clustering, PCA, KMEANS
- Performed Word Clouds for clusters

Meeshawn Marathe:

- Developed source code for the Graph-based *PageRank* algorithm:
    - Performed data parsing and created a dataframe from the list of news articles.
    - Performed sentence segmentation using nltk.sent_tokenize().
    - Performed data cleaning: removal of punctuations and conversion to lowercase.
    - Constructed an undirected graph for all the sentence nodes across 500 articles.
    - Computed *Sentence Similarity Scores* for all the sentences within an article.
    - Computed weighted sum values for each node using the Page-Rank algorithm.
- Computed Evaluation metrics for the system generated summarization by implementing ROUGE-1 score and plotted a histogram distribution of the scores across 500 news articles.
- Contributed to the making of presentation and report.

Vishal Patil

- Cleaning Dataset by removing punctuations and stopwords
- Calculated Term Frequencies and Inverse Document Frequency

- Calculated weights of each sentence by multiplying TF and IDF
- Determined top 40% of the total sentences of original text in the summary to improve performance of the algorithm
- Evaluated ROGUE-1 score for each summary

Conclusion

Page-Rank algorithm performs better compared to other algorithms, according to a comparison of the average ROUGE-1 scores computed across all the 500 news articles. Page rank, being a graph based summarization technique, requires multiple iterations in order to converge across all its nodes/sentences. Hence, its time complexity is higher compared to other algorithms and is more complex to realise compared to the other frequency based approaches. Amongst the frequency based approaches, the centroid based approach performs better compared to the standard TF-IDF sentence ranking and Luhn's method.

References

1. Dataset: CNN-DailyMail News Text Summarization | Kaggle
2. A survey on abstractive text summarization Conference Paper · March 2016. (n.d.)
3. Automatic Text Summarization Using Natural Language Processing Pratibha Devihosur1, Naseer R2 1 M.Tech. student, Dept. of Computer Science and Engineering, B.I.E.T College, Karnataka, India 2 Assistant Professor, Dept. of Computer Science and Engineering, B.I.E.T College, Karnataka, India. (n.d.).
4. Deepali K. Gaikwad, C. Namrata Mahender, "A Review Paper on Text Summarization", "International Journal of Advanced Research in Computer and Communication Engineering". Vol.5, Issue 3, March 2016.
5. Text Summarization:An Overview October 2013 Samrat BabarSamrat BabarM Tech-CseRit. (n.d.).
6. G. Vijay Kumar, M. Sreedevi, NVS Pavan Kumar, "Mining Regular Patterns in Transactional Databases using Vertical Format"."International Journal of Advanced Research in Computer Science", Volume 2, Issue 5, 2011.
7. https://iq.opengenus.org/luhns-heuristic-method-for-text-summarization/. (n.d.).
8. https://www.researchgate.net/publication/318108205_Automatic_Keyword_Extraction_for_Text_Summarization_in_Multi-document_e-Newspapers_Articles. (n.d.).
9. Text summarization using Latent Semantic Analysis August 2011Journal of Information Science 37(4):405-417 DOI: 10.1177/0165551511408848 SourceDBLP Makbule Gulcin OzsoyFerda Nur AlpaslanFerda Nur AlpaslanIlyas CicekliIlyas Cicekli. (n.d.).