Text Summarization System: An Extractive Approach using Hierarchical Text Clustering

Francisca O. Oladipo
Federal University Lokoja, Nigeria

Abdulaziz Baba-Ali Ohiani
Federal University Lokoja, Nigeria

ABSTRACT
The need for summarizing texts evolves from the large amount of data present in electronic channels which leads to distraction of users and wastage of their time. There are generally two major techniques for text summarization: extractive method and abstractive method. The extractive method has proven to be quite reliable and involves extracting the key sentences from the document to form a summary. In this paper, an unsupervised text mining model is developed for clustering and summarizing texts. The model is deployed into a web-based system for summarizing large documents. Using the informational criteria of *redundancy*, *coherence*, *speed* and *information coverage*, our approach chooses '*not likely'*, '*high*', '*fast*', and '*medium*' as semantic dimensions values for the criteria respectively.

General Terms
Natural Language Processing, Text Summarization, Text Clustering, Text Mining.

Keywords
Extractive summaries, text clustering, web application, sentence clustering.

1. INTRODUCTION
We are fortunate to live at a time where vast amount of information is readily available at the touch of a button. However, this enormous amount of information can overwhelm an individual as he can easily get lost in a sea of irrelevant information. Text summarization, the process of creating a short account of a text document which conveys the main ideas expressed in that document [1] is a branch of Natural Language Processing (NLP) that addresses this problem.

As the need of the industry for document processing increases, an increasing number of techniques for data summarization are being developed to make these documents compact and digestible [2]. Extractive and abstractive approaches are used in text summarization.

Extractive approaches transform the document into some optimal set of sentences which are easy to comprehend [3]. In extractive approaches, first, sentences are extracted from a document to a set *S*, and then, the top *k* sentences are joined to form a meaningful summary of the text [4]. This set of sentences offer much of the original ideas expressed in the document.

On the other hand, abstractive summarization uses generative techniques to produce sentences that tries to express the idea contained in a document [5]. Abstractive techniques usually require more computational resources and lots of training iterations before meaningful results can be obtained. Despite a recent focus in abstractive summarization approaches, extractive techniques have the advantage of being less complex, cheaper, and most importantly, the capability to generates grammatically correct sentences [6].

2. RELATED WORK
The significance of summarizing citation sentences to create technical summaries using "graph-based summarization model" called C-LexRank was analyzed by [7]. They discovered that technical summaries could benefit from citation sentences. While their work is geared towards citation-based summaries, our technique is more generic and tend to accommodate a broader spectrum of summaries.

Verbene *et al.* [8] developed an extractive text summarization model that can create summaries for forum threads. Their approach, however, uses post selection rather than sentence selection. They asked humans raters to rate the posts on a scale of informativeness and used this information to train their model. Interestingly, they discovered that their model produced thread summaries that were just as good (and in some cases better) than those produced by humans.

Other techniques include SVR (Support Vector Regression) model was built using annotated data set for summarizing online debate data on global warming [9]; complex network-based approach which was applied by [10]. The utilization of dynamic matrices in data summarization proved to be beneficial instead of ARD (Anti-Redundancy Detection) [11]. Moving towards cross-language summarization, [12] compressed four lingual (French, English, Portuguese, and Spanish) documents into two languages (French and English).

Few approaches for abstractive summarization were reviewed as well. One of the approaches that are most similar to the approach that this research seeks to adopt is the approach by [13], where they used a hybrid technique that consists of both semantic and statistical features to determine to score sentences. Of particular interest is the way they tried to solve the problem of redundancy by including only new sentences to the summary set if the difference between the new sentence and the sentences in the summary set is below some predefined threshold. In our research, however, we intend to solve this problem by using the Balanced Iterative Reducing and Clustering Using Hierarchy (BIRCH) algorithm which creates clusters from sentences that are similar and then select the most informative sentence from a cluster as its representative sentence, thus reducing the chances of redundancy in the final summary.

3. MATERIALS AND METHODS
Extractive text summarization produces summaries primarily by selecting a set of sentences from the document which conveys the information contained in that document.

3.1 Review of Existing Techniques
In this section, we review 3 techniques that have been used for extractive text summarization.

3.1.1 TF-ISF Algorithm

TF-ISF is a frequency-based algorithm that scores sentences based on the frequency of the words in sentences. It can handle both single-document and multi document summarization tasks.

The *TF-ISF* algorithm is made up two parts: text frequency and inverse sentence frequency. *TF* (Text frequency) value is given below in eq.1:

$$\boldsymbol{Tf}(\boldsymbol{w}) = \frac{\boldsymbol{N_w}}{\boldsymbol{N_t}} \quad (1)$$

where N_w represents the number of times a word w appears in a sentence while N_t represents the total number of words in a sentence. Conversely, the *ISF* value summarization which means Inverse Sentence Frequency) is given in eq.2:

$$\boldsymbol{Isf}(\boldsymbol{w}) = \frac{\boldsymbol{S_w}}{\boldsymbol{S_t}} \quad (2)$$

where S_w is the number of sentences that a word w appears in, while S_T represents the total number of sentences in the document. The *TF-ISF* score for a word is thus given by eq. 3.

$$\boldsymbol{Score}(\boldsymbol{w}) = \boldsymbol{Tf}(\boldsymbol{w}) * \boldsymbol{Isf}(\boldsymbol{w}) \quad (3)$$

3.1.2 Text Rank Algorithm

This approach represents the document in a graph structure. Nodes represent the sentences while edges depict the similarities among the sentences. Sentence nodes with a number of edge connections higher than some predefined threshold are more likely to selected as part of the summary set. In Figure 1 below, if the threshold is 2, then node *S2* is more likely to be selected as it has the edge connection weight of 3.

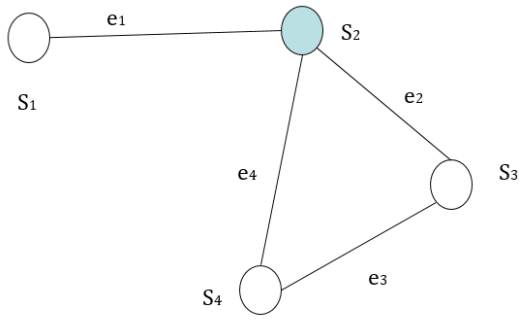

Figure 1. Graphical Representation of the Sentences

3.1.3 Latent Semantic Analysis

Latent Semantic Analysis (LSA) is an unsupervised technique for text summarization that extracts semantic representation of a document on a vector space [14]. It makes use of an algorithm called the Singular Value Decomposition (SVD) that takes an input matrix representation of a document and then decomposes it into three matrices. The problems with this technique as highlighted are its inability to handle polysemy and the fact that SVD is a slow and computationally intensive algorithm.

3.2 Proposed System

In this paper, a new system for extractive text summarization which relies on text clustering, is introduced. This approach makes use of the BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) algorithm, which is advantageous when large datasets are considered for hierarchical clustering [15].

In the proposed approach (Figure 2), the sentences are vectorized before clustering. The vectors are then clustered after which the sentence vectors are normalized. The mean value for each cluster is computed which is then used to determine the sentence that best represents a cluster by finding a sentence with a value such that the absolute value of the difference between the mean and that value is the minimum. Together these sentences that are extracted from each cluster serve as the summary for the entire document.

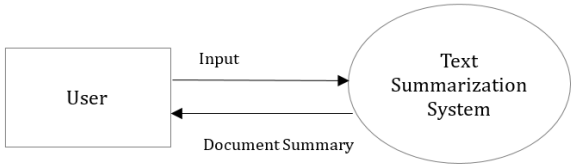

Figure 2. High-Level Modeling f the proposed system

3.3 Methodology

The first activity in the summarization pipeline is document selection. A friendly user interface was developed to make it easy to upload documents for summarization. The selected document is then parsed into raw text. Index-based sentence tokenization is carried out on the raw text in order to uniquely reference each sentence in the document.

The next step in the pipeline is case folding, where all the words in the document are converted to small letters to avoid double referencing. Afterwards, Lemmatization is carried to reduce the words into their root words. *WordNet Lemmatizer* is the tool used for this step, which is included in *nltk* module.

To reduce text noise, 'stop words' are removed. These words do not carry any semantic significance but aids in easier reading. Then the transformation of sentences into numerical values is performed. This process is known as vectorization. This was done by using the Doc2Vec module that is provided with the Genism Library.

At this stage, clustering--a very significant stage in the data summarization pipeline--is carried out. In this step, the vectors formed in the previous step, are fed into clusters. Agglomerative algorithm clusters the document by initially considering each vector a cluster. Then the vectors are normalized, and the mean of each cluster is computed. This is shown in Figure 3. A target sentence is selected from each cluster such that the absolute value of the difference between the value for that sentence and the mean of the cluster is the smallest for all sentences in that cluster.

Sentence	Vector_Value	Cluster	Cluster_Mean	Closest_Sentence_Vector
1	1.742671	0	1.768833	1.773912
2	1.746302	0	1.768833	1.773912
24	1.671568	1	1.694657	1.715157
29	3.292537	2	3.292537	3.292537
30	1.864843	3	1.841810	1.864843
32	0.010528	4	0.010528	0.010528

Figure 3. Randomly Sampled Sentences after Clustering

Finally, for each cluster, the sentences which have a smaller distance from mean are selected and merged to form a summary for the document. The standard data mining methodology that is adopted for the steps listed above is the Knowledge Discovery in Databases (KDD) methodology. This methodology is chosen because it is suitable for unsupervised data mining tasks and because of the similarities between the phases of the methodology and the steps involved in this research (Figure 4).

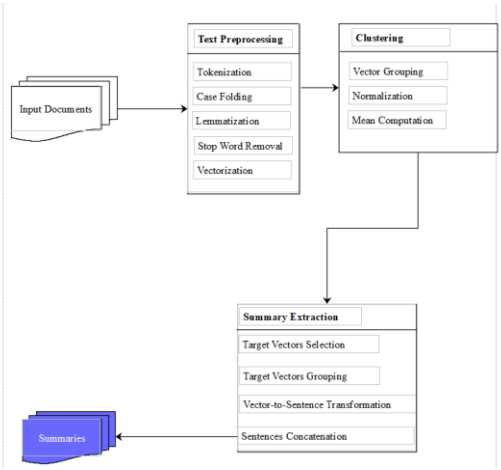

Figure 4. Text Summarization Pipeline

3.4 Software Development

The tools deployed for software development were *Bottle*, *React*, *Genism*, *Jupyter Lab*, *Axios*, and *NLTK* Library. The initialization of software development includes planning for various functionalities and features that the software would possess. Analysis of the system resulted in discovering the existing systems related to the proposed plan. Scrutinizing the results of previous systems, aids in better implementation of the proposed system.

After identifying the requirements of the system, the system is designed with the required functionalities along with its implementation. Various tools are utilized to implement the designed into working viable products. The proposed system consists of a summarization module, an API server, and a client application. These were developed and tested independently to ensure they are working properly before being integrated. The components shown in Figure 5 was tested manually using a white-box testing techniques to determine if each is working according to the specifications.

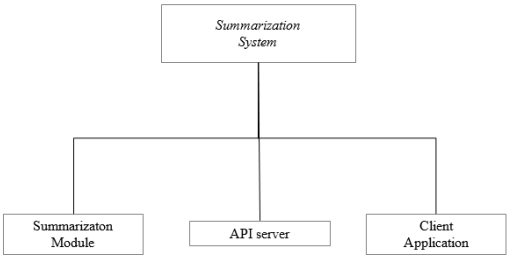

Figure 5. Modules of the Proposed Software

4. RESULTS AND DISCUSSIONS

The proposed system constitutes of the user, the API server, the client application, and the summarization module. The client application provides an interface for the users to input their data. The API server handles summarization requests and sends them the summarization module. The summarization module then performs the objection function of summarization and then sends it back through the server to the application, which lucidly present the summary. This summarization system is capable of accepting documents in multiple documents at once for summarization.

4.1 Representation of System

The graphical representation of the system is shown in figure 6.

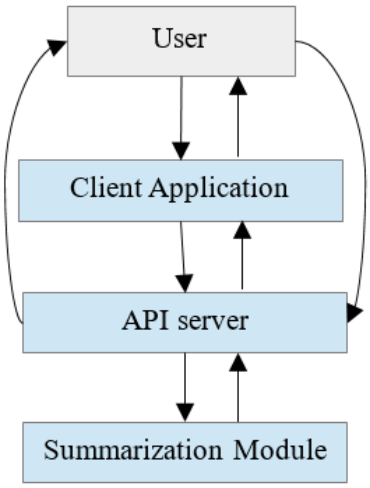

Figure 6. Illustration of the summarization system architecture

4.2 Interaction with client

This shows that how the system coordinates with the client and provide the output of their choice.

4.2.1 Input Terminal

The input physical system has two panels. One is for the uploading of documents for summarization. This panel is shown in Figure 7. Its features two option one is for uploading of the document from local disk and the other is for transfer of document over to the server for summarization. Hover on the name of the document pop up the deletes option. As the document is selected and the summarization button is selected it sends the document for summarization and users see alerts of document summarization is in progress.

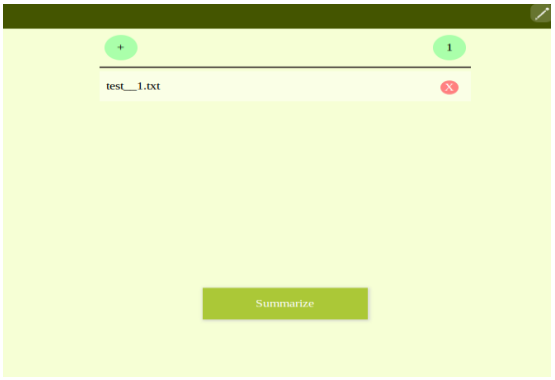

Figure 7. The Document Upload Panel

4.2.2 Output Terminal

The second panel as named above is used to present the summary of the document if it is generated along with the document title. This provides a panel for scrolling down the summary and a download option in various formats. This is depicted in Figure 8.

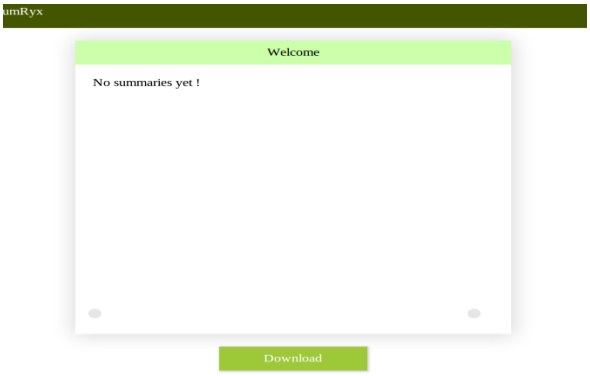

Figure 8. The View Panel for the summary of a document

Figure 9 shows the interface after a summary has been generated.

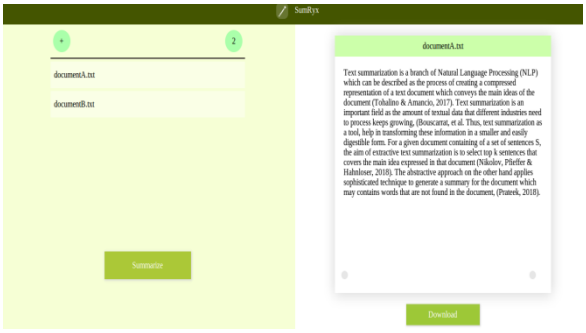

Figure 9. Summary generated by the System

4.3 Logical Representation

The logical diagram represents the interaction between the components. This depicts the flow of data and instructions among the components. This is shown in the use case diagram in Figure 10.

.

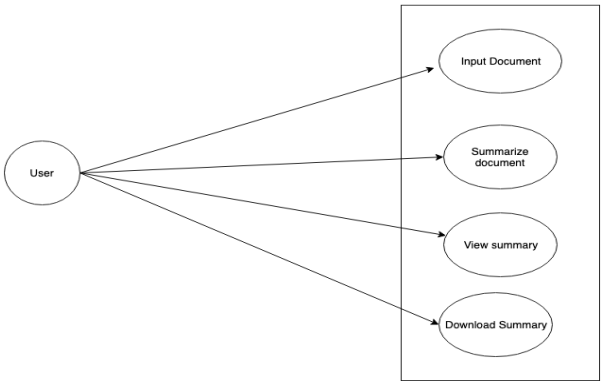

Figure 10. Use Case Diagram of the summarization system

As soon as the user enters its document into the system, it is parsed, tokenized, and cleaned. Next to the preprocessing steps, the sentences are clustered and the sentence obtaining mean near to that of the cluster is selected for summarization. Then the client may choose to view summary online or download it for offline use. These processes are depicted in Figure 11.

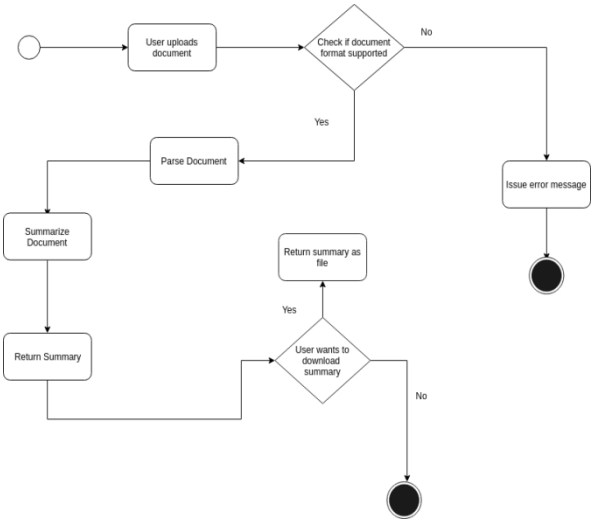

Figure 11. Activity Diagram for the Summarization System

4.4 Performance Evaluation

The proposed system for generating summary is compared with the manual summarization of document which is commonly made by authors before the submission of their books. Table 1 shows the comparison of manual summarization with the proposed system.

Table 1. Comparison of the proposed system with Manual Summarization

Metric	Manual Method	Proposed Method
Redundancy	Not Likely	Not Likely
Coherence	Very High	High
Speed	Slow	Fast

The table shows that the proposed and manual systems have similarities in redundancy and coherence. However, the proposed method is faster than the manual method when it comes to speed, but, the manual method seems to be able to

produce summaries that have higher information coverage and coherence than the proposed method.

5. CONCLUSION

An extractive text summarization model based on text clustering was developed using a hierarchical clustering algorithm called BIRCH. The model was deployed to a server application built using the bottle framework. A client application was developed to interact with the server application for summarizing documents. The client application provides a platform for users to upload documents and download/view summaries for these documents. The format for document this system accepts is only plain text. More work can be done for multi-format documents.

6. REFERENCES

[1] Tohalino, J. V., & Amancio, D. R. (2017). Extractive multi-document summarization using dynamical measurements of complex networks. Paper presented at the 2017 Brazilian Conference on Intelligent Systems (BRACIS).

[2] Rananavare, L. B., & Reddy, P. V. S. (2017). An Overview of Text Summarization. Int. J. Comput. Appl, 171(10), 1-17.

[3] Rautray, R., & Balabantaray, R. C. (2017). Bio-inspired approaches for extractive document summarization: A comparative study. Karbala International Journal of Modern Science, 3(3), 119-130.

[4] Nikolov, N. I., Pfeiffer, M., & Hahnloser, R. H. (2018). Data-driven summarization of scientific articles. arXiv preprint arXiv:1804.08875.

[5] Prateek Joshi. (2018). An Introduction to Text Summarization using the TextRank Algorithm. from https://www.analyticsvidhya.com/blog/2018/11/introduction-text-summarization-textrank-python/

[6] Nallapati, R., Zhou, B., & Ma, M. (2016). Classify or select: Neural architectures for extractive document summarization. arXiv preprint arXiv:1611.04244.

[7] Vahed, Q., Radev, D., Mohammad, S.M., Dorr, B., Zajic, D., Whidby, M., & Moon, T. (2013). Generative Extractive summaries of scientific paradigms. Journal of artificial Intelligent Research, 46, 165-201.

[8] Verberne, S., Krahmer, E., Hendrickx, I., Wubben, S., & van Den Bosch, A. (2018). Creating a reference data set for the summarization of discussion forum threads. Language Resources and Evaluation, 52(2), 461-483.

[9] Sanchan, N., Aker, A., & Bontcheva, K. (2017). Gold standard online debates summaries and first experiments towards automatic summarization of online debate data. Paper presented at the International Conference on Computational Linguistics and Intelligent Text Processing.

[10] Tohalino, V. J., Diego, R. & Amancio. (2017). Extractive multi document summarization using dynamical measurements of complex networks. arXiv:1708.01769

[11] Verberne, S., Krahmer, E., Hendrickx, I., Wubben, S., & van Den Bosch, A. (2018). Creating a reference data set for the summarization of discussion forum threads. Language Resources and Evaluation, 52(2), 461-483.

[12] Pontes, E. L., Huet, S., & Torres-Moreno, J.-M. (2018). A multilingual study of compressive cross-language text summarization. Paper presented at the Mexican International Conference on Artificial Intelligence.

[13] Yadav, C. S., & Sharan, A. (2015). Hybrid approach for single text document summarization using statistical and sentiment features. International Journal of Information Retrieval Research (IJIRR), 5(4), 46-70.

[14] El-Refaiy, A., Abas, A.R. & Elhenawy, I. (2018). Review of recent techniques for extractive text summarization. Journal of Theoritical and Applied Information Technology, 96, 7739-7759.

[15] Badry, R. M., Eldin, A. S., & Elzanfally, D. S. (2013). Text summarization within the latent semantic analysis framework: comparative study. International Journal of Computer Applications, 81(11), 40-45.