Center for Open Access in Science ▪ https://www.centerprode.com/ojit.html
Open Journal for Information Technology, 2022, *5*(2), 65-80.
ISSN (Online) 2620-0627 ▪ https://doi.org/10.32591/coas.ojit.0502.03065k

A Hybrid Model for Text Summarization Using Natural Language Processing

James Mugi Karanja & Abraham Matheka

Kenyatta University, School of Engineering, Nairobi, KENYA

Received: 7 October 2022 ▪ Revised: 20 November 2022 ▪ Accepted: 28 November 2022

Abstract

Text summarization plays an important role in the area of natural language processing. The need for information all over the world to solve specific problems keeps on increasing daily. This poses a greater challenge as data stored on the internet has gradually increased exponentially over time. Finding out the relevant data and manually summarizing it in a short time is a challenging and tedious task for a human being. Text Summarization aims to compress the source text into a more concise form while preserving its overall meaning. Two major categories of text summarization methods exist namely: extractive and abstractive. The extractive technique concentrates on determining key themes using frequency analysis of sentences in the corpus of the text. Abstractive methods write a new summary with newly generated texts which do not appear in the corpus itself. This paper presents a hybrid model for text summarization using both extractive and abstractive techniques. Term Frequency (TF) – Inverse Document Frequency (IDF) was used for extractive and T5 Transformers for abstractive summarization. Iterative Incremental Methodology was adopted in the study. The hybrid model emerged as not the best choice compared to the extractive and abstractive as it had been hypothesized in the study when the accuracy and execution time of the summary generated was considered.

keywords: extractive model, abstractive model, hybrid model, natural language processing.

1. Introduction

The volume of data online continues to grow exponentially thus pausing a great challenge on how to extract relevant information from the massive amount of data (Nguyen, Santos & Russell, 2019). The technique of text summarization has been very effective in information summarization and retrieval thus helping in time-saving while searching for relevant and critical information. This in turn helps in quick decision-making (Hassel, 2007). Text summarization is the process of using software to reduce the length of a text document to make a summary having important considerations from the original document. This is done by highlighting the most important parts of the text (Goyal, Behera & McGinnity, 2018). The input type determines the type of summarizer to be used. You can have a single document summarizer where the input is a small amount of text content or a multi-document summarizer where the input can be derived from various sources and long documents (Goyal et al., 2018). The complexity of the model to be created increases as the amount of text to be summarized rises.

Correspondence: James Mugi Karanja, Kenyatta University, School of Engineering, Nairobi, KENYA. E-mail: karanja.mugi@ku.ac.ke.

Generic summarizers treat the input without bias or prior knowledge; Domain-specific summarizers employ domain information to generate a more accurate summary based on known facts, and Query-based summarizers only contain known responses to natural language questions regarding the input text. In Query-Based Text Summarization (QBTS), a query is taken as input in this method, and depending on the query, the model can create the text's summary by choosing phrases and sentences which are closely related to the query posed as the input (Boorugu & Ramesh, 2020). The length of the input in Single Document Text Summarization (SDTS) is usually short while the length of the input in Multi-Document Text Summarization (MDTS) is usually longer and numerous documents are offered as input for summary creation. MDTS is usually more difficult than SDTS because you have to integrate multiple documents' summaries into a single document (Boorugu & Ramesh, 2020). Extractive methods create a summary from the source material by extracting important phrases, keywords, and sentences. The abstractive approaches provide a summary that resembles a human being's written abstract. The extractive method assures that the created summaries are grammatically and semantically correct, while the abstractive method generates more diverse and new content (Yao et al., 2018).

2. Problem statement

The need for information all over the world to solve specific problems keeps on increasing daily. This poses a greater challenge as data stored on the internet has gradually increased exponentially over time (Nguyen et al., 2019). The structure of how processes used to be executed in various industries has also changed due to the onset of the COVID-19 epidemic. In the education sector, it has led to the adoption of E-learning in most Institutions of Higher Learning as students are encouraged to get access to research and learning materials online instead of visiting physical libraries. With the vast amount of data online, extracting meaningful information might not be easier (Goyal et al., 2018). This might lead to a lot of time utilized during research and learning as students go through various learning materials online. This calls for a solution that can help transform the huge amount of data into summarized information that is easily consumable. Research previously conducted concentrated on the summarization of only plain text. This calls for the development of a hybrid model for text summarization that will be able to extract and summarize text from various sources, i.e., pdf, plain text, and website.

3. Literature review

This chapter analyzes related works, techniques for text summarization, and various models in use.

4. Text summarization

The process of creating a summary necessitates first reading and comprehending the original content. The main components of the paper are stated based on the known events, facts, or situations to satisfy the summary's objective. The summary would not include all of the material in the original text, but only the parts that were judged relevant. This is self-evident, given that the summary's objective is to decrease the quantity of information in the source papers. The summary is subsequently prepared in a suitable output format after defining the main aspects of a document. In general, there are three components to summarization: input, analysis, and output. Humans often require a comprehension of the native language in which the text material is written. The goal of the summary and the target audience would both need to be determined throughout the analysis. The final step before the summary is delivered to the user would be to create a proper output format for the summary (Radev & Erkan, 2015).

There are several things to consider regardless of whether the summarizer is a person or a machine. In terms of input: The summarizer would have to select how to approach reading the material based on how it is structured. For example, information relevant to the analysis stage may be found in the headers of chapters or the labels of Figures and Tables. Some document metadata, such as keywords in HTML websites, may also be useful. If the document was classed and the summarizer has access to the class or domain to which it belongs, it may be feasible to use domain-specific information to help in the analysis and output phases. The summarizer may be influenced by the language used in the text documents. Human summarizers normally need to know the language in which the material was written (Yao et al., 2018).

5. Related Works

Every day, massive volume of information is published on the internet (Boorugu & Ramesh, 2020). This necessitates the use of a solution that can assist in the transformation of large amounts of data into summarized information. This necessitates the application of a text summarizing technique, which aids in the reduction of massive material into key points. The method works by keeping important information while producing a condensed version of the text.

Automatic summarization (Chu, Song & Jaimes, 2015) or indexing (Garg, Hassan & Chaudhury, 2015) has been investigated in a variety of study disciplines, including video, audio, and document (text) processing. Text summarizing techniques were employed to construct concise summaries of documents. A frequency thresholding method was developed in an early text summarizing study. The study used term frequency, inverse document frequency (TF-IDF) method (Shimada, Okubo, Yin & Ogata, 2018), which has been found to attain a reasonable degree of performance, has been introduced to improve frequency-driven approaches. TF-IDF concentrates only on extractive text summarization where it uses word frequency to generate a summary. Even though some words might have a higher frequency, they might not be the main points thus leading to a misleading summary.

Eberts et al. (2015) suggested a system for automatically condensing educational video content. Their method pinpoints the exact moment and location in video footage where presentation slides appear. In their method, they combine image processing and machine learning approaches. They also created electronic lectures and screencasts with the technology. The findings suggested that the summaries created gave viewers more information. The research concentrates on video content and fails to address summarizing text content which is usually used in most institutions.

A method for summarizing oral lectures was proposed by Chen et al. (2011). On a graph created, a random walk is performed making use of automatically extracted key phrases and latent semantic analysis with probabilities in their method. They used their method to obtain each document summary from lecture documents using the extractive summary generation method (Chen et al., 2011). The researcher failed to elaborate more on how to extract the content from various sources to generate the summary. Li et al. (2014) suggested a completely automatic approach for capturing the full presentation utilizing camera techniques such as panning, tilting, and zooming to extract the semantic structure of a typical academic lecture video (*Ibid.*, 2014). General video summarizing approaches were shown to obtain more precise display structures than their system. This concentrates on video content summary but doesn`t talk of text content summary generation which is easily accessible to a large population.

It is sometimes suggested that studying ahead of time for a class is critical for students to familiarize themselves with keywords from the study, and learn new concepts and terminology. Shimada et al. (2018) stressed the necessity of giving students a glimpse of what they will study ahead of time. Furthermore, effective preparation before lectures start has been linked to

improved understanding and grasping of the concepts taught during the lecture. Students are frequently requested to study a textbook or preview content to prepare for their next class at university (Shimada et al., 2018). There is need of having a way of creating summarized content that can be used in study previews instead of going through a whole topic in a textbook which will lead to time-saving among the learners.

6. Hybrid summarization

Combining more than one feature of text summarization leads to the development of a hybrid model. Rani et al. (2017) proposed a hybrid model which makes use of the word frequency and the location of the paragraphs. Words contained in paragraphs at the beginning of the document are given more weight than words in the preceding paragraphs. Even though the model developed is hybrid, it only makes use of extractive technique of text content summarization.

MuraliKrishna et al. (2013) offer a hybrid summarizing system in which sentences are extracted from documents based on the sentence scoring method. The average of the values evaluated using statistical and linguistic methodologies is used to calculate the sentence-scoring method. The duplicate information in these retrieved sentences is handled using an iterative clustering process. The final result, known as the document summary, provides the most significant sentences in the document related to the query without redundancy. The generated sentences can be sorted by their sentence score or the order in which they appeared in the source. The researcher makes use of extractive method of summary generation and doesn't mention abstractive technique.

A hierarchical hybrid multi-document model using extractive technique was invented by (Celikyilmaz & Hakkani-Tur, 2010). The model is designed with the capability of splitting a document into major subtopics which in turn are divided into more small topics. The content in the subtopics is summarized independently via extractive technique and the results are combined to form a final summary. The outcome of this system would be more fine-tuned if they would have been channeled to a model using abstractive technique. Even though the research developed a hybrid model, it only made use of one major technique of content summarization.

A multiple-text document system that uses hybrid summarization techniques was developed by (Dave & Jaswal, 2016). The system uses extractive techniques in the initial stage and the output is channeled to an abstractive model which uses Word Graph generation which locates important nodes from the extractive summary using heuristic rules. Heuristics rules aid in easier problem-solving and provide a shortcut to solving difficult problems but don't necessarily give an optimum solution. This might affect the output of the summary. The system only tackles the summarization of text content pasted on it and doesn't provide the allowance of summarization of online text content, i.e., from Wikipedia. Instead of using Word Graph, the research proposes the use of T5 transformers for the abstractive summary generation to increase the accuracy of the output summary and also provide the allowance of getting the text content to be summarized from various sources, i.e., extracted from pdf documents and website URL (Uniform Resource Locators) like Wikipedia.

7. Text summarization models

7.1 *Term Frequency (TF) – Inverse Document Frequency (IDF)*

The TF-IDF algorithm was utilized in the extractive summary generation. Term Frequency (TF) helps to count the frequency of words. The frequency discovered is used to assess the word's significance. The more frequently a word appears in the source document, the more important it is. The straightforward explanation for TF is that it counts the frequency with which

a word emerges or is seen in a document (Meena et al., 2020). IDF provides unique words with a higher value and repeated words with a lower value. TF occasionally overestimates the significance of stop words depending on how often they appear. Inverse Document Frequency determines the rarest of words that appear in the document to address TF's issue. IDF is the inverse of TF; when the two are combined, the result is TF-IDF refers to the product of TF and IDF. TF–IDF is a powerful algorithm for summary generation though it is only applicable for extractive summary generation and can't rewrite the summary using different words other than the one in the main document (Meena et al., 2020).

8. Text rank algorithm

The text rank is an unsupervised method that uses weights as a value to rate sentences. The foundation of the text rank algorithm may be traced back to page rank on Google system, which performs the ranking of websites regarding their links and their significance (Mihalcea, 2004). As the name implies, a directed graph is built using phrases. This is referred to as the graph-based ranking method. The phrases are referred to as nodes or vertices, and edges are used to connect nodes that are related (Mallick et al., 2019). The text rank algorithm is a recommender-based system in which the vertices joined by the edges recommend the relevance of the phrases in the graph. The weights assigned to the sentences are used to rank the sentences and the summary is generated from the sentences having more weight. The text Rank Algorithm employs only the extractive method of summary creation. A system employing both extractive and abstractive summary generation models can be of greater help in gauging the validity of the summary created.

9. Sequence to sequence model

Time-Frequency Representation Summary (TFRS) method employs an abstract summary model known as a sequence-to-sequence model to improvise new words while maintaining the meaning of the original text. It uses an encoder-decoder model which translates sequences of varying lengths as input and output (Song, Huang & Ruan, 2019). The encoder-decoder component's Long Short-Term Memory (LSTM) is useful for capturing long-term dependencies. The training and inference phases of the encoder-decoder model are separated. In the training and inference phases, both the encoder and the decoder are utilized. Sequence-to-sequence models have provided feasible solutions for abstractive summarization but are still hard to tackle long text dependency in the summarization task (Liao, Zhang, Chen & Zhou, 2020).

10. Transformers for text summarization

Transformer is an encoder-decoder model and converts all NLP problems into a text format. They make use of Transfer machine learning where pre-trained models are used to perform different tasks thus providing higher performance (Gupta, Chugh, Anjum & Katarya, 2021).

Due to the large amount of data used to train the transformers; they usually create well optimized models thus making them to be most preferred.

A research conducted by Gupta et al. (2021) on various types of transformers for text summarizations portrayed that T5 transformer outperformed the other types of transformers as shown in the table below.

Models	Evaluation Metrics		
	ROUGE-1	***ROUGE-2***	***ROUGE-L***
Pipeline - BART	0.38	0.28	0.38
BART modified	0.40	0.28	0.40
T5	**0.47**	**0.33**	**0.42**
PEGASUS	0.42	0.29	0.40

Figure 1.

11. Methodology

An Iterative Incremental Methodology was adopted in the system development. Several testing and reviews were conducted at each step thus enabling system specifications to be error-free and reliable. Testing and debugging were conducted on smaller iteration hence making the task easy and effective compared to testing complete system requirements at once. In the beginning, simpler implementations of a hybrid model for text summarization were designed and developed. Additional functionalities were incorporated into the system in various iterations. The design modification was made and new functional capabilities were added in every cycle. The iterations continued until the development process was completed. This methodology enabled the user to evaluate the system functionality periodically until the final product was delivered. This created room for capturing new requirements and implementing them.

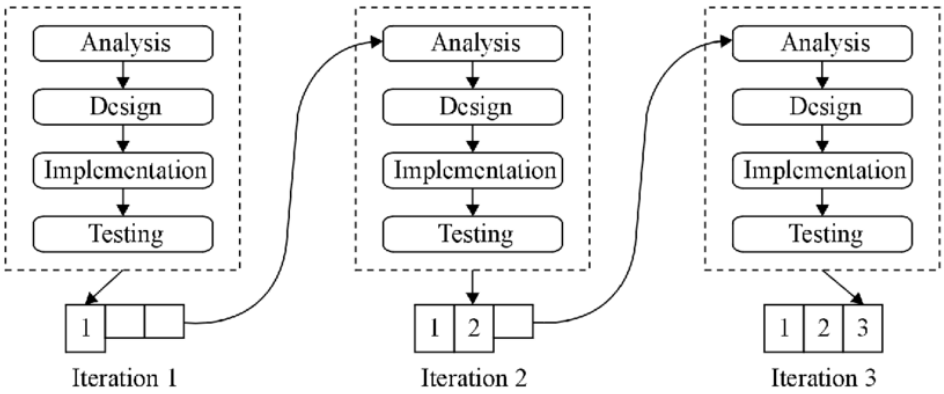

Figure 2. Iterative incremental methodology

12. Planning phase

The hybrid model for text summarization should be able to summarize text content from a pdf document, plain text pasted on the system, and text content scrapped from the website URL (Uniform Resource Locators). Extractive text summarization was conducted first and the output channeled to the abstractive model to generate a hybrid summary.

13. Analysis

In this phase, the specifications of the hybrid model for text summarization were studied based on the problem that had been identified in the planning phase. Analysis was conducted to choose the best logic, database models and to identify any other requirements. The

system architecture was stipulated in this stage. Term Frequency (TF) – Inverse Document Frequency (IDF) and T5 Transformers were to be used in the development of a hybrid model for text summarization.

14. Design and development

The design of a hybrid model for text summarization prototype was produced in this phase. The requirements captured in the previous phases were used to develop the system. An automatic text summarization model was developed in this phase. The model had the capability of summarizing contents extracted from pdf, Wikipedia, and raw text. Python programming language was used in the model backend development. CSS and HTML were used for front-end development. Streamlit framework was used in the front-end development of the model. Streamlit framework helped turn python codes into web apps in a very short time for free and no front-end experience was required. Building an app was conducted with a few lines of code and simple API calls. Widgets were added easily like declaring variables and no backend code was required to describe routes, handle HTTP web requests, either connect a frontend, draft HTML, CSS, and JavaScript. The model adopted both abstractive and extractive summarization techniques.

15. Summary creation process

Data Preprocessing was conducted to convert the data into a machine-readable form of the vector. The process started with the Tokenization of Sentences. The text was divided into sentences. This was implemented via the use of a sentence tokenizer from the NLTK toolkit in Python. Once the paragraphs were divided into sentences, all special characters and stop words were removed. It's conceivable that the text contains some characters that aren't needed. All of the characters that are not needed were eliminated. Word Tokenization was conducted using word space where each of the article's phrases was broken down into words. After the word tokenization, each word's weighted occurrence frequency was determined and then used to generate an extractive summary. The output from the extractive summary was keyed into the T5 transformer model to generate the hybrid summary.

16. Testing phase

Testing began after the current build iteration had been developed and implemented to find and track any potential defects or problems that may have existed in the model. System testing was carried out for each iteration to determine if all the user requirements were captured and implemented. The hybrid model for text summarization was tested to check if it met the research objective. The plain text was pasted on the system and submitted to create a summary for both the abstractive and extractive options. This step was repeated for pdf content and plain text extracted from a website URL (Uniform Resource Locators) to make sure that the research objective was achieved.

17. Evaluation phase

The Iterative life cycle ended at this stage. If there were bugs and requirements not met in the testing stage, the development was subjected to iterations. If no bugs were found, the hybrid model for text summarization was deployed for use. Once the hybrid model for text summarization passed the testing stage, it was ready for deployment in production.

18. Implementation

Python programming language was used in the model backend development while CSS and HTML were used for frontend development.

19. Results

To get started, a dashboard panel was created to act as the user interface for the project. Streamlit Platform is used in the creation of the graphical user interface. This is conducted by importing all the required packages and then creating the GUI (Graphical User Interface).

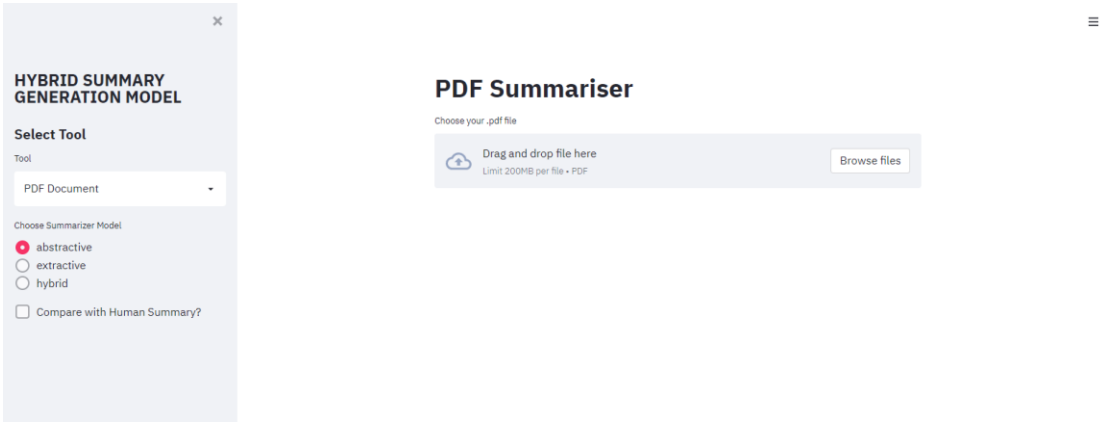

Figure 3. Dashboard

20. Extractive summary model

This summarizer model constructed summary by selecting relevant sentences derived from the original text. The TF-IDF algorithm was utilized in the extractive summarization. Term Frequency (TF) helped to count the frequency of words. The frequency discovered was used to assess the word's significance. The more frequently a word appeared in the source document, the more important it was. The straightforward explanation for TF is that it counted the frequency with which a word emerged or seen in a document. IDF provided unique words with a higher value and repeated words with a lower value. TF occasionally overestimated the significance of stop words depending on how often they appeared.

21. Abstractive summarization model

The abstractive model created its phrases and sentences to provide a more comprehensive summary, similar to the one human being developed. We made use of a transformer network that relied primarily on many levels of attention. For memorizing the sequence of words in the input sequence, it didn't employ RNN and instead relied on attention layers and positional encoding. The global dependencies formed by using several attention layers aided in the parallelization of input processing. Encoder and decoder layers were coupled to a multi-head attention layer and feed-forward network levels in the transformer model. The model used cosine and sine functions to recall the location and sequence of words, resulting in positional encoding. The encoder and decoder layers used a multi-head attention layer and applied a mechanism called self-attention. The input was keyed into 3 linked layers to generate query (Q), key (K), and value (V) vectors. The vectors were subdivided into n vectors (Gupta et al., 2021).

22. Hybrid summarization model

This model was developed by combining the extractive model with the abstractive model. The output of the extractive model was used as the input for the abstractive model thus generating a hybrid summary.

22.1 *Hybrid model rouge score results*

Results from online content (Wikipedia)

	rouge1	rouge2	rougeL
Recall	0.5556	0.1132	0.3333
Precision	0.1515	0.0305	0.0909
F1-Score	0.2381	0.0480	0.1429

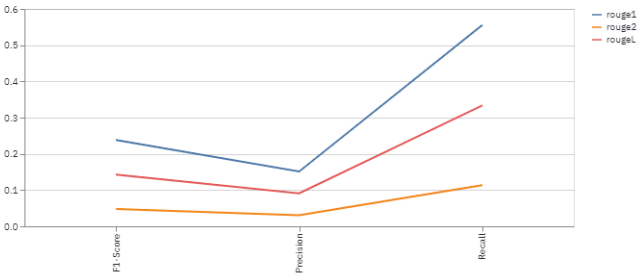

Figure 4. Hybrid results from online content (Wikipedia)

This graph shows results for the summary generated using a hybrid model with an online content source of data. The values for recall, precision, and f1-score for rouge1, rouge2, and rougeL are displayed in the table and a graph is drawn to show their relationship. Rouge 1 has the highest values followed by rouge2 and rougeL has the least values. A very big difference is noted in the values for recall, precision, and f1-score across the 3 rouges (rouge1, rouge2, rougeL). This is portrayed by the line graphs for the rouge score being away from each other.

22.2 *Extractive rouge score results*

Results from online content (Wikipedia)

	rouge1	rouge2	rougeL
Recall	0.4938	0.2264	0.2375
Precision	0.3990	0.1827	0.1919
F1-Score	0.4413	0.2022	0.2123

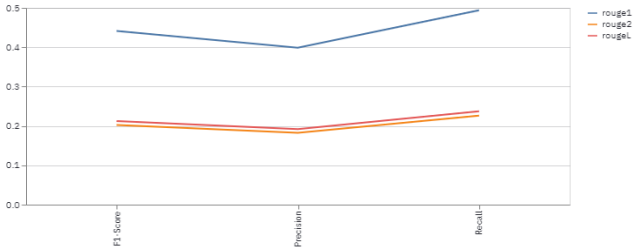

Figure 5. Extractive results from online content (Wikipedia)

This graph shows results for the summary generated using an extractive model with online content source of data. The values for recall, precision, and f1-score for rouge1, rouge2, and rougeL are displayed in the table and a graph is drawn to show their relationship. Rouge1 has the highest values followed by rouge2 and rougeL has the least values. A very big difference is noted

in the values for recall, precision, and f1-score for rouge1 while the scores for rouge2 and rougeL have a slight difference. This is portrayed by the line graphs for the rouge1 score being away from the rest while the line graph for rouge2 and rougeL being closer to each other.

22.3 *Abstractive rouge score results*

Results from online content (Wikipedia)

	rouge1	rouge2	rougeL
Recall	0.9434	0.8077	0.8491
Precision	0.2525	0.2132	0.2273
F1-Score	0.3984	0.3373	0.3586

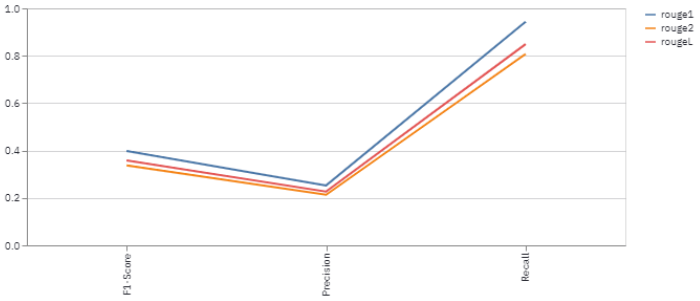

Figure 6. Abstractive results from online content (Wikipedia)

This graph shows results for the summary generated using an abstractive model with online content source of data. The values for recall, precision, and f1-score for rouge1, rouge2, and rougeL are displayed in table and a graph is drawn to show their relationship. Rouge1 has the highest values followed by rouge2 and rougeL has the least values. A small difference is noted in the values for recall, precision, and f1-score across the 3 rouges (rouge1, rouge2, rouge L). This is portrayed by the line graphs for the rouge score being closer to each other.

23. Discussion

Table 1. Recall average scores

		HYBRID	**EXTRACTIVE**	**ABSTRACTIVE**
ROUGE 1	Online	0.5556	0.4938	0.9434
	Pdf	0.5333	0.3822	0.9434
	Text	0.6	0.3618	0.8103
	TOTALS	**1.6889**	**1.2378**	**2.6971**
	AVERAGE	**0.562966667**	**0.4126**	**0.899033333**
ROUGE 2	Online	0.1132	0.2264	0.8077
	Pdf	0.1364	0.0705	0.8077
	Text	0.1111	0.0596	0.6667
	TOTALS	**0.3607**	**0.3565**	**2.2821**
	AVERAGE	**0.120233333**	**0.118833333**	**0.7607**
ROUGE L	Online	0.3333	0.2375	0.8491
	Pdf	0.3556	0.2102	0.8491
	Text	0.3455	0.1908	0.7586
	TOTALS	**1.0344**	**0.6385**	**2.4568**
	AVERAGE	**0.3448**	**0.212833333**	**0.818933333**

From the results in table 1, the Abstractive model has the highest average score for Rouge 1, Rouge 2, and Rouge L followed by the Hybrid model and finally the Extractive model. If we consider the recall ability of the models, it is clear that the Abstractive model emerges as the best model followed by the Hybrid model and the last one is the Extractive model. The summary generated by the abstractive model has more similarity with the one generated by a human being as compared to the summary generated by the other models.

Table 2. Precision average score

		HYBRID	**EXTRACTIVE**	**ABSTRACTIVE**
ROUGE 1	Online	0.1515	0.399	0.2525
	Pdf	0.1212	0.303	0.4967
	Text	0.1667	0.2778	0.2374
	TOTALS	**0.4394**	**0.9798**	**0.9866**
	AVERAGE	**0.146466667**	**0.3266**	**0.32886667**
ROUGE 2	Online	0.0305	0.1827	0.2132
	Pdf	0.0305	0.0558	0.2132
	Text	0.0305	0.0457	0.1929
	TOTALS	**0.0915**	**0.2842**	**0.6193**
	AVERAGE	**0.0305**	**0.094733333**	**0.206433333**
ROUGE L	Online	0.0909	0.1919	0.2273
	Pdf	0.0808	0.1667	0.2273
	Text	0.096	0.1465	0.2222
	TOTALS	**0.2677**	**0.5051**	**0.6768**
	AVERAGE	**0.089233333**	**0.168366667**	**0.2256**

Table 2 shows the precision score for all the models. From this table, the Abstractive model was found to have the highest precision score for Rouge 1, Rouge 2, and Rouge L, while the Extractive model was the second and the Hybrid model had the lowest precision score. If we choose a model in terms of precision score, the Abstractive model was the one recommended.

Table 3. F1-Score average

		HYBRID	**EXTRACTIVE**	**ABSTRACTIVE**
ROUGE 1	Online	0.2381	0.4413	0.3984
	Pdf	0.1975	0.338	0.3984
	Text	0.2609	0.3143	0.3672
	TOTALS	**0.6965**	**1.0936**	**1.164**
	AVERAGE	**0.232166667**	**0.364533333**	**0.388**
ROUGE 2	Online	0.048	0.2022	0.3373
	Pdf	0.0498	0.0623	0.3373
	Text	0.0478	0.0517	0.2992
	TOTALS	**0.1456**	**0.3162**	**0.9738**
	AVERAGE	**0.048533333**	**0.1054**	**0.3246**
ROUGE L	Online	0.1429	0.2123	0.3586
	Pdf	0.1317	0.1859	0.3586
	Text	0.1502	0.1657	0.3438
	TOTALS	**0.4248**	**0.5639**	**1.061**
	AVERAGE	**0.1416**	**0.187966667**	**0.353666667**

Table 3 shows F1-score calculated from all the models. From this table, the Abstractive model had the highest F1 score, followed by the Extractive model and finally, the Hybrid model

had the least score. If we were to consider the F1-score, the Abstractive model would be the best choice.

The results for rouge score were used to help in choosing the best-performing model. From the result, there was a very slight or no difference in the rouge score when we consider various sources of inputs (pdf, plain text, and online). This implied that the source of data/input doesn`t affect the quality of the summary generated and also the precision and recall ability of the model used.

The abstractive model was found to have higher recall ability as compared to the other models. It was followed by the Hybrid model and the last one was the Extractive model.

The Abstractive model had the best average precision as compared to all the other models. It was followed by the Extractive model and the last one was the Hybrid model.

On F1-score, the Abstractive model had the best average F1-score, followed by the Extractive model and the Hybrid model had the lowest average F1-score.

24. Execution time

Table 4. Execution Time results

EXECUTION TIME IN SECONDS					
	Online	**Pdf**	**Plain Text**	**Total**	**Average**
Abstractive	80.8308	94.428	67.0551	**242.3139**	**80.7713**
Hybrid	86.7931	95.0698	68.4162	**250.2791**	**83.42636667**
Extractive	2.7926	21.5619	0.8644	**22.4263**	**7.475433333**

From Table 4, the Extractive model took the least time to generate the summary then followed by the abstractive model, and finally, the hybrid model took more time as compared to the extractive and abstractive models. If we consider the time taken for execution, the extractive model becomes the best choice followed by the abstractive model.

25. Models Ranking

Table 5. Models ranking results

ROUGE SCORE & EXECUTION TIME MODELS POSITION RANKING							
	Recall	**Precision**	**F1-Score**	**Execution Time**	**Total**	**Average**	**Position**
ABSTRACTIVE	1	1	1	2	5	1.25	1
HYBRID	2	3	3	3	11	2.75	3
EXTRACTIVE	3	2	2	1	8	2	2

Table 5 displays the rank position of all the models in regards to recall, precision, f1-score, and execution time. The models were ranked depending on the position they hold. The positions were summed up and then averaged to get their overall position. From the research outcome displayed in Table 5, the abstractive model held position 1, the extractive model position 2, and the hybrid model position 3.

26. Conclusion and future works

26.1 *Conclusion*

The study hypothesis was that the hybrid model would have the best results as compared to the separate extractive and hybrid models. From the research findings, each of the models was found to be having its strengths and drawbacks. When considering the recall ability of the models, the abstractive model was found to be the best followed by the hybrid model, and lastly the extractive. The results in terms of precision of the models portrayed the abstractive model to be the best, followed by the extractive model, and finally, the hybrid model was the least. From the f1-score, the abstractive model emerged first, the extractive the second, and the hybrid model the last.

The models' execution time was noted down and used to make an analysis. From the results obtained, the extractive model took the least time to generate the summary followed by the abstractive model and the hybrid model.

The performance of the models generated from the rouge score helped in ranking the models. The abstractive model emerged to have the best results as compared to the extractive model and hybrid model. If the model choice is made depending on the rouge score, the abstractive model was the best choice. The models' performance based on time execution portrayed the extractive model as the best choice followed by the abstractive model. When we find the average score positions from the rouge score and timely execution, the Abstractive model became the leading one followed by the extractive model and the hybrid model was the last. Therefore, the hybrid model isn't the best choice compared to extractive and abstractive as it had been hypothesized in the study when the accuracy and execution time of the summary generated is considered.

26.2 *Future works*

The research was based on text summarization thus more research should be conducted on the automatic summarization of graphical content and video content. It has also not been easy to summarize research papers effectively especially summarizing papers with scientific formulas, equations, graphs, and tables. A tool needs to be established which will be able to understand contents from equations, formulas, graphs, and tables for effective summarizing of scientific papers.

Acknowledgements

This research did not receive any specific grant from funding agencies in the public commercial, or not-for-profit sectors.

The authors declare no competing interests.

References

Boorugu, R., & Ramesh, G. (2020). A survey on NLP based text summarization for summarizing product reviews. *Proceedings of the 2nd International Conference on Inventive Research in Computing Applications, ICIRCA 2020*, 352-356. https://doi.org/10.1109/ICIRCA48905.2020.9183355

Celikyilmaz, A., & Hakkani-Tur, D. (2010). A hybrid hierarchical model for multi-document summarization. *ACL 2010 - 48th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, (July), 815-824.

Chen, Y.-N., Huang, Y., Yeh, C.-F., & Lee, L.-S. (2011). Spoken lecture summarization by random walk over a graph constructed with automatically extracted key terms. *Twelfth Annual Conference of the International Speech Communication Association*.

Chu, W.-S., Song, Y., & Jaimes, A. (2015). Video co-summarization: Video summarization by visual co-occurrence. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3584-3592.

Dave, H., & Jaswal, S. (2016). Multiple Text Document Summarization System using hybrid Summarization technique. *Proceedings on 2015 1st International Conference on Next Generation Computing Technologies, NGCT 2015*, (September), 804-808. https://doi.org/10.1109/NGCT.2015.7375231

Eberts, M., Ulges, A., & Schwanecke, U. (2015). Amigo-automatic indexing of lecture footage. *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, 1206-1210.

Garg, R., Hassan, E., & Chaudhury, S. (2015). Document indexing framework for retrieval of degraded document images. *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, 1261-1265.

Goyal, P., Behera, L., & McGinnity, T. M. (2018). A context-based word indexing model for document summarization. *IEEE Transactions on Knowledge and Data Engineering*, *25*(8), 1693-1705. https://doi.org/10.1109/TKDE.2012.114

Gupta, A., Chugh, D., Anjum, & Katarya, R. (2021). *Automated news summarization using transformers*. Retrieved from http://arxiv.org/abs/2108.01064.

Hassel, M. (2007). *Resource lean and portable automatic text summarization*. https://www.csc.kth.se/utbildning/forskar/avhandlingar/doktor/2007/HasselMartin.pdf.

Li, K., Wang, J., Wang, H., & Dai, Q. (2014). Structuring lecture videos by automatic projection screen localization and analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(6), 1233-1246.

Liao, P., Zhang, C., Chen, X., & Zhou, X. (2020). Improving abstractive text summarization with history aggregation. *Proceedings of the International Joint Conference on Neural Networks*. https://doi.org/10.1109/IJCNN48605.2020.9207502

Mallick, C., Das, A. K., Dutta, M., Das, A. K., & Sarkar, A. (2019). Graph-based text summarization using modified textrank. In J. Nayak, A. Abraham, B. M. Krishna, G. T. Chandra Sekhar & A. K. Das (Eds.), *Soft computing in data analytics* (pp. 137-146). Singapore: Springer Singapore.

Meena, S. M., Ramkumar, M. P., Asmitha, R. E., & Emil Selvan, G. S. (2020). Text summarization using text frequency ranking sentence prediction. *4th International Conference on Computer, Communication and Signal Processing, ICCCSP 2020*, 0-4. https://doi.org/10.1109/ICCCSP49186.2020.9315203

Mihalcea, R. (2004). *Graph-based ranking algorithms for sentence extraction, applied to text summarization*. (4), 20-es. https://doi.org/10.3115/1219044.1219064

MuraliKrishna, V. R., Pavan, Kumar, Y. S., & Satyananda, R. C. (2013). A hybrid method for query based automatic summarization system. *International Journal of Computer Applications*, *68*(6), 39-43. https://doi.org/10.5120/11587-6925

Nguyen, H., Santos, E., & Russell, J. (2019). Evaluation of the impact of user-cognitive styles on the assessment of text summarization. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, *41*(6), 1038-1051. https://doi.org/10.1109/TSMCA.2011.2116001

Radev, D. ., & Erkan, G. (2015). Proceedings of Document Understanding Conference Workshop. *The University of Michigan at Duc*, 120–127.

Rani, S. S., Sreejith, K., & Sanker, A. (2017). A hybrid approach for automatic document summarization. *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 663-669. https://doi.org/10.1109/ICACCI.2017.8125917

Shimada, A., Okubo, F., Yin, C., & Ogata, H. (2018). Automatic summarization of lecture slides for enhanced student preview-technical report and user study. *IEEE Transactions on Learning Technologies*, *11*(2), 165-178. https://doi.org/10.1109/TLT.2017.2682086

Song, S., Huang, H., & Ruan, T. (2019). Abstractive text summarization using LSTM-CNN based deep learning. *Multimedia Tools and Applications*, *78*(1), 857-875.

Yao, K., Zhang, L., Du, D., Luo, T., Tao, L., & Wu, Y. (2018). Dual encoding for abstractive text summarization. *IEEE Transactions on Cybernetics*, *50*(3), 985-996. https://doi.org/10.1109/TCYB.2018.2876317


C O A S