Legal Case Text Summarization

Abstract

In the recent decade, Natural Language Processing has made noteworthy progress and several tools and techniques have been developed that provide high-quality summaries. In this paper, we take into consideration the UCI legal corpus and apply preprocessing techniques to it i.e., cleaning and tokenization. Next, we applied KL (Kullback Lieber) Divergence, SumBasic, BERT, Neural Network based Sentence Extraction and Text Rank algorithms to generate summaries of the available legal cases. We use ROUGE to measure the overlap between the model-created summaries and human-authored summaries.

1 Introduction

The volume of legal cases is increasing exponentially, making it challenging for legal practitioners to keep track of all the cases and derive insights from them. One solution to this problem is text summarization, a technique that helps to condense lengthy legal case texts into shorter summaries while retaining the essential information. With the advancements in Natural Language Processing (NLP) and machine learning, several tools and techniques have been developed to generate high-quality summaries.

This research paper focuses on the application of KL Divergence, SumBasic, BERT, Neural Network based Sentence Extraction and Text Rank algorithms for text summarization of legal cases. The UCI legal corpus, consisting of 3315 legal cases, is taken as the data-set for experimentation [1]. The KL Divergence algorithm is a statistical measure that helps to quantify the difference between two probability distributions, SumBasic is a simple yet effective algorithm that selects the most important sentences for summarization, BERT is a deep learning-based algorithm that utilizes pre-trained models to generate high-quality summaries, and Text Rank is a graph-based algorithm that identifies the most significant sentences in a text.

To evaluate the accuracy of our summarization techniques, we use the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metric, which measures the overlap between the model-created summaries and human-authored summaries [2]. We expect our results to demonstrate the effectiveness of using citation in legal case summarization, as well as provide insights into the most effective NLP techniques for this task. Overall, our study aims to contribute to the growing body of research on NLP-based legal case summarization and advance the development of more efficient and accurate tools for legal professionals.

The rest of the paper is organized as follows. Section 2 provides the Corpus Evalution and Preprocessing. Section 3 presents the Methodology. Section 4 discusses the Model Implementation, Section 5 discusses the evalution methods, section 6 descibes the experimental analysis and results, followed by a conclusion in Section 7.

2 Corpus Evaluation and Preprocessing

Corpus evaluation and preprocessing are critical steps when preparing a dataset for any NLP task, especially text summarization. The corpus has to be carefully evaluated for quality and preprocessed to remove any irregularities that could cause poor model performance. The preprocessing stage carefully focuses on identifying any existent errors, inaccuracies, or inconsistencies that could adversely affect the model outcomes.

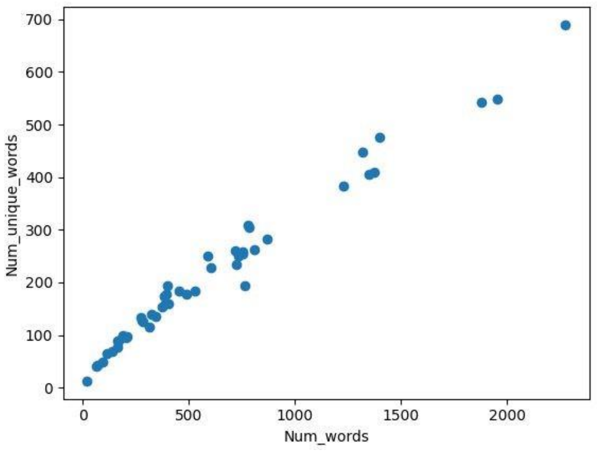


Figure 1: Num. of Unique Words vs.Words

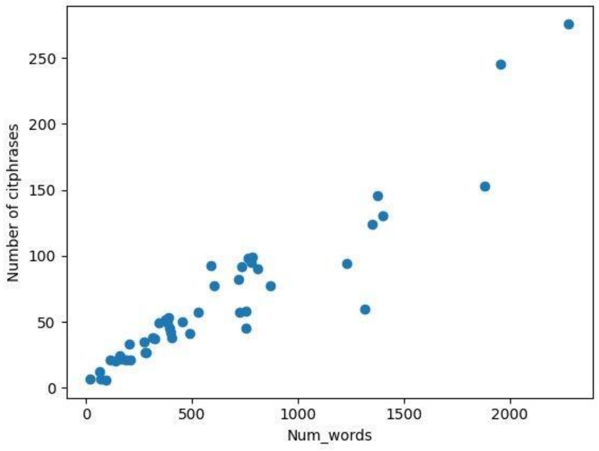


Figure 2: Num of citphrases vs. Words

The legal case reports dataset was chosen from the UCI machine learning repository. The original case files contained XML files imported and read into a pandas data frame where each row signifies a different case and the information variables are sentences, citphrases, catchphrases, and AustLII citations amongst others.

The XML files were imported using the ElementTree module in the Python programming language. The ElementTree module parses through each XML file and extracts the necessary information such as case names, catchphrases, sentences, and cite phrases. To cleanse and preprocess the dataset a regular expression pattern was used to remove all characters that are not letters or whitespaces. This expression aims at removing punctuation marks and non-whitespace characters. Next, it aims and replacing multiple consecutive whitespaces with a single whitespace character.

3 Methodology

For evaluating this legal corpus we use a combination of extractive and abstractive methods. Extractive Methods: KL Divergence, SumBasic, and Text Rank are all extractive summarization methods. Extractive summarization involved identifying the most important sentences or phrases from the original citphrases text[3]. This approach typically focuses on ranking sentences based on their relevance and similarity.

- SumBasic Algorithm:

  For this study, we employed the SumBasic Algorithm for extractive summarization. SumBasic is a graph-based algorithm that works by creating a network of sentences based on their similarity.

- KL Divergence (Kullback-Leibler Summarization):

  This algorithm also uses a graph-based approach which uses Kullback Leibler divergence measure to computer the similarity between sentences. This one incorporates a similar approach to the SumBasic algorithm. Once implemented it creates a graph of sentences where the node represents the sentences, and the edges represent the similarity between them.

- Text Rank Algorithm:

  The Text Rank algorithm is rooted in Google's PageRank algorithm that is used for ranking webpages and returning them for the searches made through its engine. Page Rank ranks a web page based on its importance, usage, and frequency of linkages to other web pages. The Text Rank algorithm, similar to the other two, uses graph representation with the nodes representing the sentences and edges representing the relationship between them. The weight on the edges defines the level of similarity existent between the node or sentences. Popularly the weight for these is calculated using cosine similarity between two vectors as we have done.Once the importance is calculated the algorithm iterates over the nodes and edges until converges to a stable point.

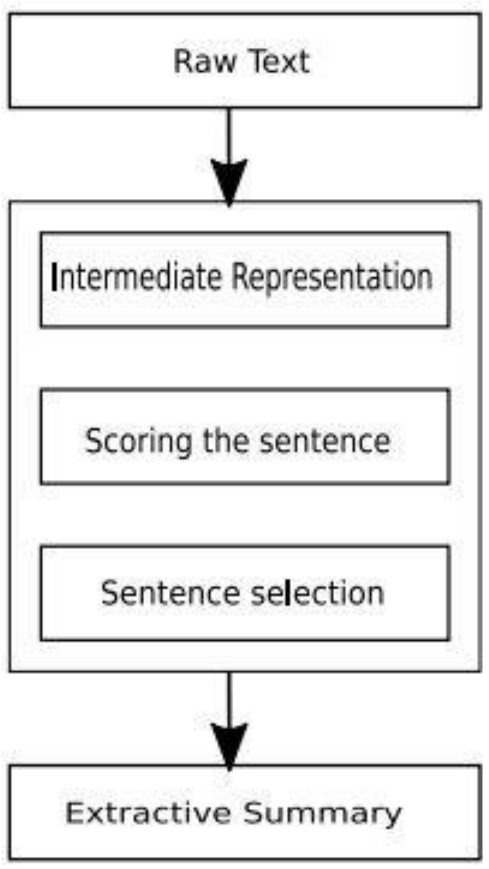


Figure 3: Extractive Summarization Flow

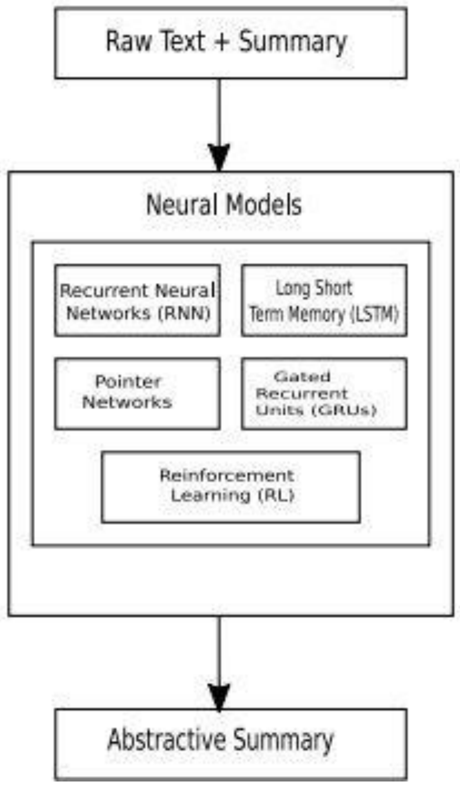

Raw Text + Summary
Neural Models
Recurrent Neural Networks (RNN)
Long Short Term Memory (LSTM)
Pointer Networks
Gated Recurrent Units (GRUs)
Reinforcement Learning (RL)
Abstractive Summary

Figure 4: Abstractive Summarization Flow

- NN-SE (Neural Network-based Sentence Extraction):

  NN-SE is another extractive method used for text summarization. NN-SE, a deep learning method was proposed by Nallapati et al. in 2017 and uses the convolutional neural network to identify the important sentences in a document. A document is given as input and NN-SE uses pre-trained word embeddings to vectorize each sentence present. Once vectorized, these sentences are passed through CNN which classifies each of them as important or unimportant. NN-SE has been proven to achieve state-of-art performance for several data-set and has created a benchmark for itself in text summarization.

Abstractive Methods: Abstractive summarization is a natural language processing technique that generates a summary not exactly based on the phrases found in the document. Abstractive methods work in contrast to the extractive which selects sentences or phrases from the document given. Abstractive methods are focused on capturing the important information from an input and creating a relevant summary that is close to a human version of the same[4]. In our project, we have used BERT to summarize each legal case citphrases document to generate a meaningful summary.

- BERT (Bidirectional Encoder Representations from Transformers):

  Abstractive summarization using BERT is a recent technique and it utilizes transformer-based language models. BERT is a deep learning model that is pre-trained and performs at state-of-art benchmarks. In this method, the model is trained on a large corpus for it to train itself on the domain of the data and learn summarization[5]. It is trained alongside the human-generated summaries for that corpus and that enhances its output. It learns to identify the more relevant words and phrases that capture and identify the main essences of the document. Another feature of BERT is that it creates summaries that have a better flow and sentences structure and looks realistic as compared to abstractive methods

4 Model Implementation

we applied five different algorithms for legal case text summarization: SumBasic, KL Divergence, TextRank, Neural Network (NN), and BERT to generate summaries for legal cases.

- SumBasic Algorithm:

  We used a legal case dataset and extracted the catchphrases and citation phrases from each case. We then applied the SumBasic algorithm to the citation phrases to generate a summary that consists of the most important sentences in the document. The algorithm works by calculating the word frequency of each sentence and selecting the sentences with the highest word frequency to include in the summary.

  The algorithms first calculate the frequency of each word in a document using the ‘calculate word frequency’ function and then it scores each sentence by summing the frequencies of each word and dividing it by the total number of words present in the sentence. This is done using the ‘calculate sentence scores’ function. After doing so the sentences with the highest scores are selected depending on the desired size of the summary using the ‘sumbasics ummarize’ function.

After running this algorithm our dataset now contained some newly added features that include 'num words','num unique words', 'num citphrases','citphrases summary', 'catchphrases summary', and the 'smabasic summary'. To evaluate the performance of the SumBasic algorithm, we compared the generated summaries to the given catchphrase summaries for each case. We measured the quality of the generated summaries by calculating the number of overlapping words between the generated summary and the catchphrase summary.

- KL Divergence:

  This model is based on the idea of comparing the probability distribution of words in a given sentence with the distribution of words in the entire document. The strategy behind this approach is that the more different a sentence is compared to the overall documents, the more it is likely to contain valuable information.

  In the model implementation, the code begins by defining a 'calculate-word-frequency' function, which calculates the frequency of each word in the document. It uses the 'word-tokenize' method from NLTK and a default dictionary to store word frequencies. Next, the 'calculate-word-distribution' function is defined, which calculates the total number of words in the documents and the probability distribution of each word referring to the frequency dictionary created in the previous step. This information is used in the 'calculate-sentence-scores' method that tokenizes each sentence and calculates its word frequency and word distribution. It calculates the KL Divergences between the sentences and the word distributions and summarizes the KL score for each word to get the final KL divergence score of the sentence. Lastly, the 'klsum-summarize' function tokenizes the given document and calculates word frequency and distribution using the functions defined prior. Then it calculates the divergence score for each sentence and selects the top sentences based on scores to return the summary.

  KL divergence model is applied to a legal case study dataset, and the resulting summaries are stored in the kl-div-citphrases-summary column of the data frame. To evaluate the effectiveness of this model, the summaries are compared to the given catchphrase summaries in the catchphrase's summary column.

- TextRank

  In the Text Rank algorithm, the similarity between two sentences is computed using cosine similarity which measures the cosine of the angle between two vectors representing the sentences. The greater the similarity between the two sentences the closer the cosine similarity value to 1. For our project, we used the Text Rank algorithm to summarize the given legal corpus cases by extracting the most important sentences. The implementation initiates with normalizing whitespaces in the text and checks if there is an empty string or one that contains only whitespaces. Next, it calculates the cosine similarity.

  The Text Rank method implemented has several attributes including the damping coefficient that enhance the summarization quality by determining the weight that is assigned to the PageRank calculation, the convergence threshold which determines when the algorithm has converged, and the iteration steps.

  The 'sentence-similarity' calculates the similarity between the two sentences using cosine similarity, build-similarity-matrix which builds the similarity matrix for the input sentences, run-page-rank which runs the PageRank algorithms on the similarity matrix, get-sentence, which returns the sentence at a given index and get-top-sentences which returns the top sentences from the PageRank Calculation.

The implementation aims at summarizing a given document running the Text Rank to extract the most important sentences returned as the output summary.

- BERT

  BERT (Bidirectional Encoder Representations from Transformers) BERT is a pre-trained transformer-based neural network architecture that generates contextualized word embeddings. To further enhances the quality of our summarization results we implemented a BERT-based summarization model considering it is shown to achieve state-of-the-art performance for a variety of NLP tasks including summarization.

  We have used the 'bert-extractive-summarizer' library to implement the model and it allows us to fine-tune a pre-trained model on our legal corpus to generate summaries that our tailors to our specific use case.

  Similar to how we processed the data for all the algorithms, we concatenated the input text i.e. the citation phrases into a single document and then created a summary close to the length of our reference summary i.e. the associated catchphrases summary.

- Neural Network based Sentence Extraction

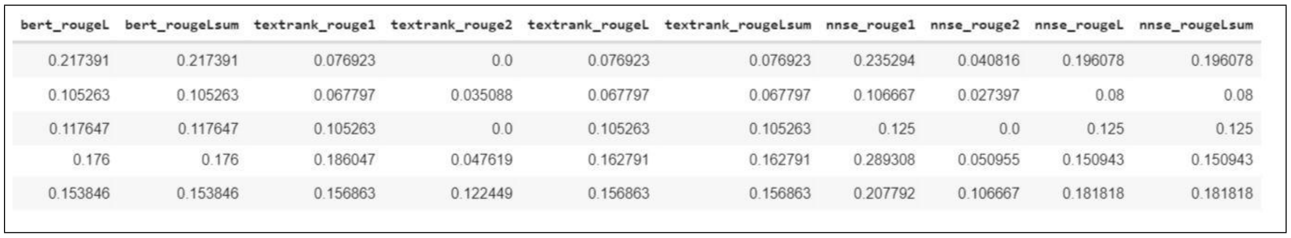
  The Neural Network-based Sentence Extraction (NN-SE) model is a powerful tool for summarizing text. In this project, we implement the NN-SE model using the BERT-base-uncased pre-trained model and the transformers library.

  The NN-SE model extract sentences that are relevant to the theme and makes a list of those sentences to create a summary consisting of the topmost relevant sentences depending on the length of the output defined. In our project, we created summaries equivalent to the size of the reference summary in order to have a better comparison. To generate the summary, like any other model the input sentences are tokenized and the resulting tensors are passed though BERT. Next a softmax function is applied to the logits output from the model and the sentences with highest probabilities are selected. The selected sentences are returned as the summary.

  Our implementation of the NN-SE model is evaluated on a legal corpus dataset, where it is compared to other summarization models. We find that the NN-SE model performs well in generating summaries that are concise and accurately capture the main points of the input document. Overall, the NN-SE model is a promising approach for summarizing text and can be used in a variety of applications, including document summarization and automated content generation.

5 Evaluation Methods

we evaluate the performance of five different text summarization algorithms, namely KL Divergence, SumBasic, BERT, Text Rank, and NNSE. We use the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metric to measure the quality of the generated summaries in comparison to the human-authored summaries.



bert_rougeL	bert_rougeLsum	textrank_rouge1	textrank_rouge2	textrank_rougeL	textrank_rougeLsum	nnse_rouge1	nnse_rouge2	nnse_rougeL	nnse_rougeLsum
0.217391	0.217391	0.076923	0.0	0.076923	0.076923	0.235294	0.040816	0.196078	0.196078
0.105263	0.105263	0.067797	0.035088	0.067797	0.067797	0.106667	0.027397	0.08	0.08
0.117647	0.117647	0.105263	0.0	0.105263	0.105263	0.125	0.0	0.125	0.125
0.176	0.176	0.186047	0.047619	0.162791	0.162791	0.289308	0.050955	0.150943	0.150943
0.153846	0.153846	0.156863	0.122449	0.156863	0.156863	0.207792	0.106667	0.181818	0.181818

Figure 5: ROUGE scores of five text summarization algorithms (KL Divergence, SumBasic, BERT, Text Rank, and NNSE) evaluated for legal case summarization.

ROUGE is a commonly used metric in NLP for evaluating the quality of text summarization systems[7]. It uses recall to measure the overlap between the generated summary and the human-authored summary. ROUGE provides different variants such as ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-Lsum, which measure the overlap at different levels, ranging from unigrams to longer sequences of words.

To compute ROUGE scores for our summarization models, we use the Python package 'rouge-score'. Before computing ROUGE scores, we preprocessed the generated summaries and human-authored summaries by removing all non-alphanumeric characters and extra whitespaces using regular expressions. We then passed these preprocessed summaries to the 'rouge-score' function, along with the reference summary, to obtain the ROUGE scores.

6 Experimental Analysis and Results

In this research paper, we conducted an experiment on a legal case summary dataset and implemented five different algorithms for summarization: Sum Basic, KL Divergence, TextRank, BERT, and NN-SE. We then evaluated the performance of each algorithm using four different evaluation metrics: ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-Lsum.

Our results showed that the BERT algorithm achieved the highest ROUGE scores for all four evaluation metrics, indicating that it produced the most accurate and informative summaries. The NN-SE algorithm also performed well, achieving second place in all metrics.

On the other hand, the Sum Basic algorithm achieved the lowest ROUGE scores for all four metrics, indicating that it produced the least informative and accurate summaries. The KL Divergence and TextRank algorithms also performed relatively poorly compared to BERT and NN-SE.

We also analyzed the relationship between the number of citation phrases in the summary and the number of words in the summary. Our linear graph showed that as the number of citation phrases in the summary increased, the number of words in the summary increased. This suggests that the use of citation phrases is an effective strategy for summarization, as it allows for the inclusion of important information in a concise manner.

Overall, our experiment showed that BERT and NN-SE are promising algorithms for legal case summarization, while Sum Basic, KL Divergence, and TextRank may not be suitable for this task. Our results and analysis can serve as a guideline for researchers and practitioners in the field of legal case summarization to choose the appropriate algorithm and strategies for their specific needs.

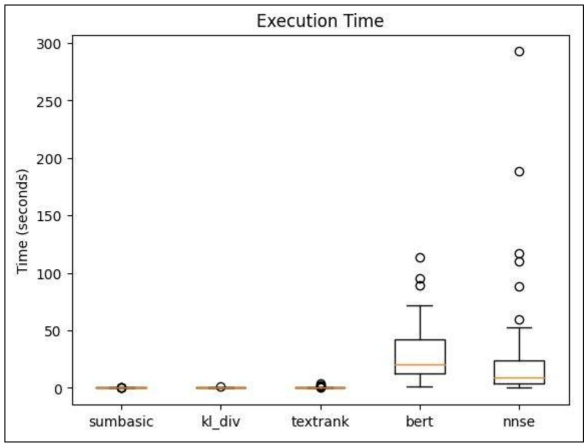


Figure 6: Model Type vs Execution time for sample cases

Table 1: Results Table

Model Name	**ROUGE 1**	**ROUGE 2**	**ROUGE L**	**ROUGE Lsum**
Sum Basic	0.166759	0.049400	0.147180	0.147180
KL Divergence	0.128869	0.034337	0.086350	0.086350
TextRank	0.143723	0.038076	0.124859	0.124859
BERT	0.279789	0.070220	0.177350	0.177350
NN-SE	0.221577	0.043209	0.155671	0.155671

7 Conclusion

he UCI legal corpus is a rich dataset. The 3500+ cases repository contains cases of various intensities and domains, and it was slightly tedious to import and cleanse the dataset considering its size and complexity. The dataset contains XML files that are imported, converted, and moved to a data frame.

In conclusion, in this project, we aimed to explore and compare various text summarization techniques both extractive and abstractive methods on the legal corpus. The five techniques that were used are TextRank, NN-SE, SumBasic, KL Divergence, and BERT.

Our results show that BERT and NN-SE outperformed the remaining techniques in terms of generating high-quality summaries that are able to capture the main idea of each case. TextRank and SumBasic also produced reasonable results but weren't as accurate as BERT and NN-SE. KL Divergence seemed to perform with the lowest accuracy and quality of the summary.

The Rouge method was applied to the results produced by each algorithm by comparing them to their respective reference summary available. Although the performance of each model can be clearly differentiated, we also have to take into account the quality of the input text given.

The take-away message of this research project is that automatic text summarization can be a useful tool for legal professionals who need to review large volumes of legal documents quickly and efficiently. However, it is important to choose the right technique and parameters for each specific task to ensure the best results.

We believe that the models implemented are useful for low-resource summary creation, par-

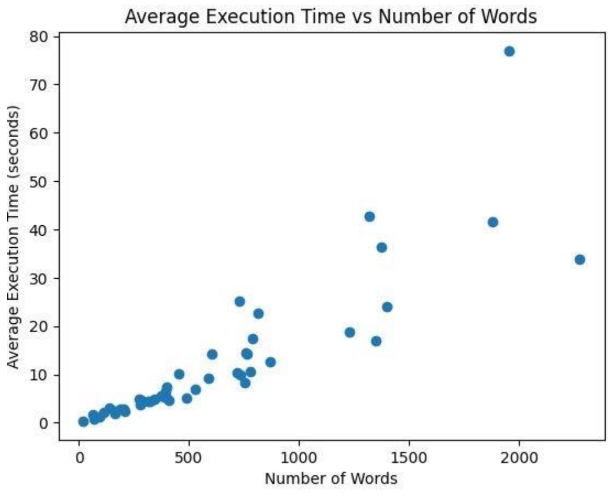


Figure 7: Average Excution Time vs Number of Words

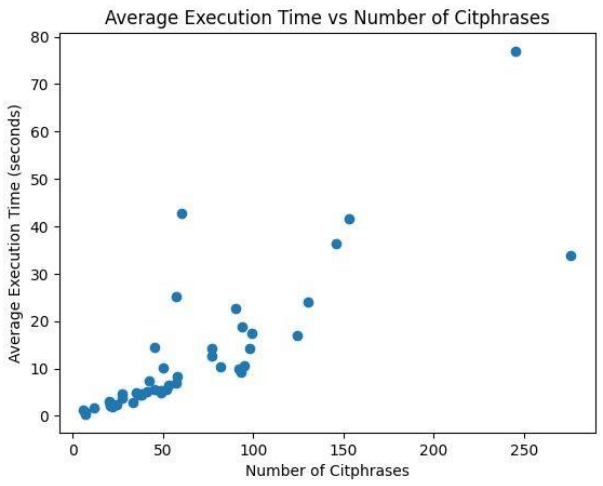


Figure 8: Average Excution Time vs Num of citphrases

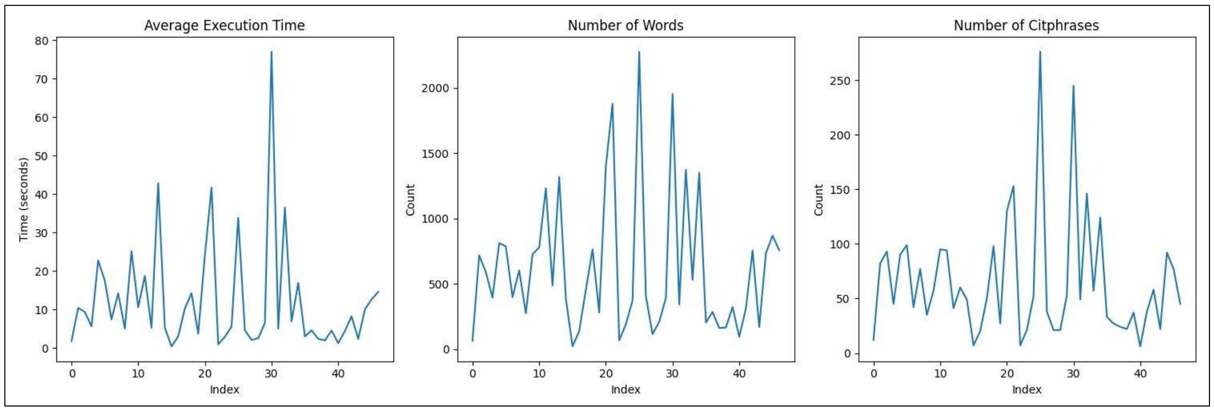


Figure 9: Average execution time vs number of words and citphrases in case document

ticularly in areas with limited internet or mobile phone connectivity. The models we developed can easily be deployed to provide summary information for individuals who may not have access to a large amount of data. In the future, we plan to explore ways to improve the performance of these models and investigate the use of large language model transformers to further enhance the accuracy and efficiency of text summarization. Overall, our project highlights the potential of text summarization technology to facilitate access to information and improve communication in a variety of contexts, including legal documents, news articles, and academic papers

References

[1] F. Galgani, P. Compton, and A. Hoffmann. Citation based summarisation of legal texts. In PRICAI 2012, volume LNCS 7458, pages 40â€"52. Springer, Heidelberg, 2012.

[2] F. Galgani, P. Compton, and A. Hoffmann. Towards automatic generation of catchphrases for legal case reports. In the 13th International Conference on Intelligent Text Processing and Computational Linguistics, volume 7182 of Lecture Notes in Computer Science, pages 415â€“426, New Delhi, India, 2012. Springer Berlin Heidelberg.

[3] F. Galgani and A. Hoffmann. Lexa: Towards automatic legal citation classification. In J. Li, editor, AI 2010: Advances in Artificial Intelligence, volume 6464 of Lecture Notes in Computer Science, pages 445 â€“454. Springer Berlin Heidelberg, 2010.

[4] F. Galgani, P. Compton, and A. Hoffmann. Combining different summarization techniques for legal text. In Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data, pages 115â€“123, Avignon, France, April 2012. Association for Computational Linguistics.

[5] F. Galgani, P. Compton, and A. Hoffmann. Knowledge acquisition for categorization of legal case re- ports. In D. Richards and B. Kang, editors, PKAW 2012, volume LNAI 7457, pages 118â€“132. Springer, Heidelberg, 2012.

[6] Deepali Jain , Malaya Dutta Borah, Anupam Biswas, Summarization of legal documents: Where are we now and the way forward, Elsevier: Computer Science Review, March 2021.