Legal Data Assistive Tool Using Deep-Learning

Sheetal Takale*, Shantanu Payal†, Shubham Jagtap‡, Pranav Jagtap§, Aftab Khan¶,
Department of Information Technology,
Vidya Pratishthan's Kamalnayan Bajaj Institute of Engineering & Technology, Baramati
Pune, Maharashtra, India
Email: *sheetaltakale@gmail.com, †shantanupayal23@gmail.com, ‡shubham.jagtap2000@gmail.com,
§jagtapspranav24@gmail.com, ¶aftab.khan.it.2512@gmail.com

***Abstract*—The proposed work aims to develop legal document summary using abstractive summarization techniques, with a focus on creating informative and concise summaries of legal documents. Modern deep neural network based pretrained models such as Distilled Bi-directional and Auto-regressive Transformer(DistillBart), Legal Pegasus, and Legal Longformer Encoder-Decoder(LED) are used to deal with the problem of low training dataset availability. Various sentence ranking approaches such as keywords, Parts Of Speech(POS) tagging, chunking, the position of sentences in the document and sentence centrality based on similarity are used to deal with the issue of large size of legal judgements. To improve informativeness of summary, focus is on entities such as important dates, concerned people, and referred acts in the legal case. The performance of different models employed for summarization are evaluated using evaluation metrics such as Bilingual Evaluation Understudy(BLEU), Precision, Recall, F1 Score, and Recall Oriented Understudy for Gisting Evaluation(ROUGE). The proposed system is valuable for legal professionals to quickly understand the main points of a legal document and make informed decisions.**

***Index Terms*—legal document summarization, abstractive summarization, sentence ranking, POS tagging, chunking, sentence similarity, DistillBART, Legal Pegasus, Legal LED**

I. Introduction

In recent years, the need for effective summarization techniques has increased, especially in the field of legal cases and judgements. Legal document summaries help lawyers, judges, and other legal professionals to quickly get to the heart of a case. Summarization of legal documents can be performed using either the Extractive Summarization technique or the Abstractive Summarization technique [1], [2]. In Extractive summarization, detailed extracts, such as the key phrases or sentences, are select from the source documents, which are further used to build the summary. Whereas, the abstractive summarization generates new sentences that precisely convey the same meaning.

The proposed work focuses on creating summaries of legal documents using abstractive summarization technique. Task of abstractive summarization of legal documents is carried out using different Deep Learning Transformer models such as DistilBART, Legal Pegasus, and Legal LED. All the models are based on the basic Seq2Seq Encoder-Decoder architecture. We have employed various approaches for ranking based on keywords, POS tagging, chunking based approach,position of sentence in the document and sentence similarity in order to generate the best possible summary.For improving the informativeness of summary our proposed approach identifies special entities, such as dates, people, and acts in the legal case. Performance evaluation of all the proposed approaches is carried out using BLEU, Precision, Recall, F1 Score, and ROUGE. These are commonly used evaluation metrics for text summarization tasks.

II. Literature Survey

Legal professionals must read through lengthy documents to find instances that are significant as precedents in ongoing litigation. Summarising the legal case records is a time-consuming task. Summarizing Indian legal documents is difficult because legal documents are lengthier and more complex than other documents. Another major issue is that the training datasets are smaller because it costs money to hire professionals to prepare summaries. For abstractive summarization, the Sequence-to-Sequence model with Encoder-Decoder architectures that use RNNs is the most widely recognised framework. However, for Indian legal document abstractive summarization, length of document and availability of training dataset becomes the bottelneck.

Modern Deep Neural Network based abstractive summarization methods include, CNN / Daily Mail based Pointer-Generator [3], pre-trained BERT model based abstractive summarization technique:BERTSumAbs [4], Pegasus [5], BART [6], and Longformer [7]. However, the input token limit of the majority of abstractive summarization models is typically lower than the length of legal case documents. Bajaj et al. [8] built a two-stage extractive-abstract technique for summarising lengthy documents. For summarising them, Gidiotis and Tsoumakas [9]suggested a "Divide- and-Conquer" strategy. LegalSumm [10] is the method for legal document abstractive summarization. However, Only 200 tokens may be generated using LegalSumm, which is considerably fewer than the desired summaries.

Due to unavailability of dataset of Indian Legal Documents Summarization, a novel methodology using two text summarization models BART and PEGASUS without the need for a dataset was proposed by Satyajit Ghosh et.al. [11]. The abstractive summarization approach for Indian Legal Text Summarization [11] includes collecting Indian legal documents, extracting texts using Optical Character Recognition (OCR), removing noise, normalizing text, constructing dictionaries, and applying models to summarize the documents.

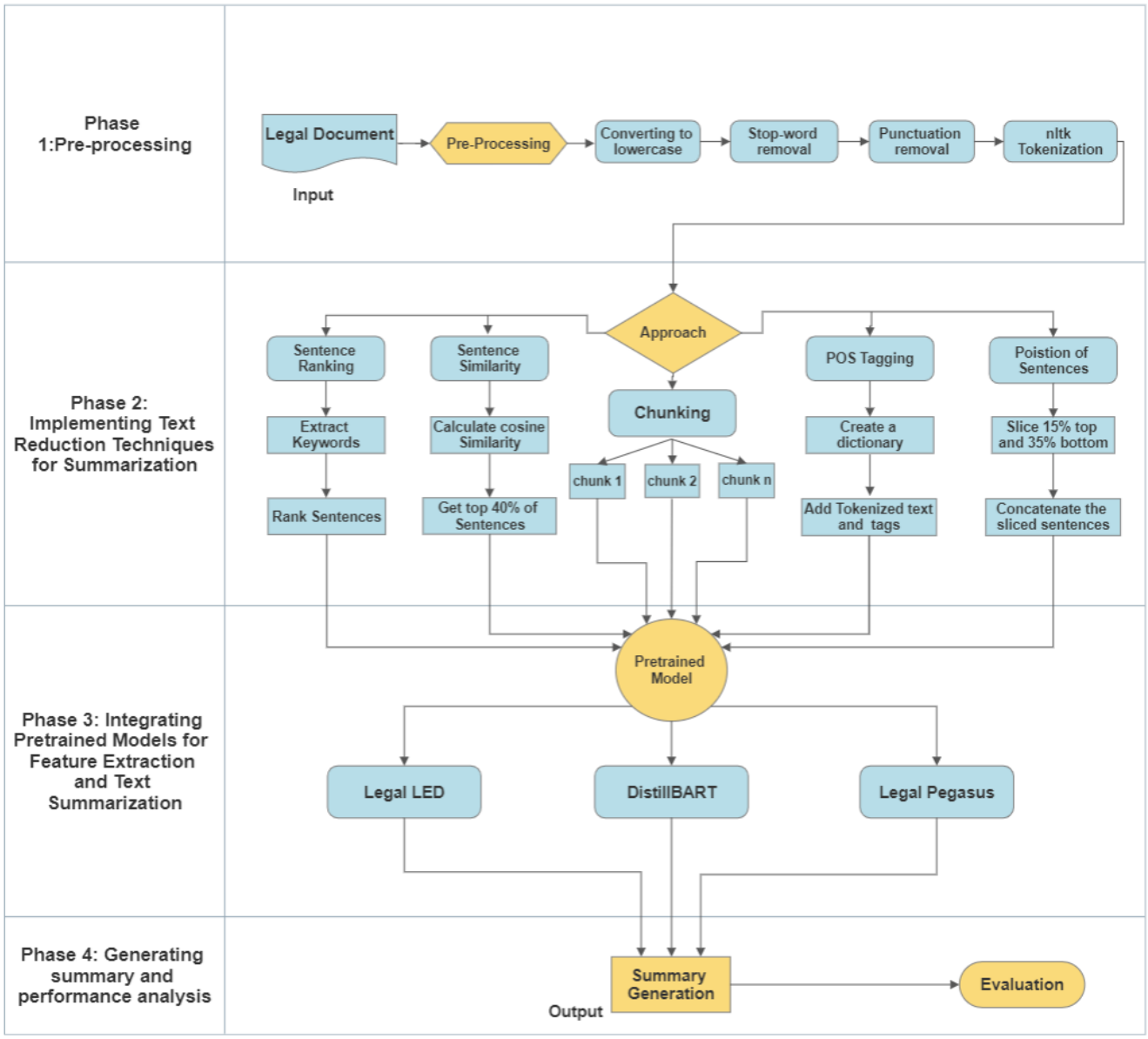

Fig. 1. System Architecture

Sequence-to-sequence Models for Abstractive Summarization of Dutch Court Judgements [12] involves a reinforcement learning and a deep learning. Different datasets were used for different models,for BART-CNN/Daily mail, for RL-Rechtspraak, for BART-Rechtspraak. The results of the paper showed that BART performed better than the RL model using ROUGE scores and other evaluation metrices.

The use of Document Context Vector and Recurrent Neural Networks in a Seq2Seq-based generative model for abstractive and extractive text summarization [13] is presented. It suggests using document-context in Seq2Seq models, much like how humans understand the text by reading the title and abstract to learn essential elements.

A Hybrid strategy is presented for automated text summarising of legal cases. [14] A three-step method for text summarising of court cases is proposed, consisting of pre-processing, clustering using K-means clustering of comparable lines, and extraction of the best sentences from each cluster using TF-IDF vectorizer is used.

The paper [15] discusses the usage of a neural network design known as the Transformer model for sequence-to-sequence transduction, including machine translation. It exclusively relies on attention mechanisms and does not use recurrent or convolutional neural networks. The Transformer connects the encoder and decoder and allows for global correlations among input and output sequences by using a self-

attention mechanism.

Algorithm 1 : Sentence Ranking with Keywords

1: Input the legal document text.
2: Pre-process the input text and tokenize to sentences, tokenize to words, remove stop words.
3: Define a function to extract keywords:

```
def extract_keywords():
    keywords = []
    keywords = nltk.FreqDist(keywords)
    keywords = keywords.most_common(200)
```

4: Define function to rank the sentences based on the number of keywords present in each sentence:

```
def rank_sentences(sentences, keywords):
  sentence_scores = {}
  for sentence in sentences:
    score = 0
    for keyword in keywords:
      if keyword in sentence.lower():
        score += 1
     sentence_scores[sentence] = score
     sorted_sentences =
           sorted(sentence_scores.items(),
           key=lambda x: x[1], reverse=True)
  return [sentence[0]
           for sentence in sorted_sentences]
```

5: Generate the summary using the pre-trained seq2seq model.
6: Evaluet and Output the generated summary.

III. Dataset

Dataset Description: The data set is collected through Legal Search of Manupatra Legal Search System (access provided by GNP Legal). Data set used in this research consists of 45 legal documents belonging to five different domains or subjects such as civil, banking, consumer, education and human rights in Bombay High Court and Supreme Court of India. These judgments in the dataset are 2020 onwards. Manupatra uses human legal experts to generate Case Note or the abstractive Summary of Judgement. In addition it also includes information regarding: Petition No., Appellant, Respondent, Coram, Counsels, Subject, Catch Words, Mentioned in, Acts/Rules/Orders, Disposition, and Decision. The gold standard case summary for each of these cases is obtained from Manupatra. To deal with the low data availability, we have used pretrained Transformer models.

IV. Methodology

An encoder-decoder architecture is used in deep neural network transformer models for sequence transduction. A series of symbol representations (x1,..., xn) are entered into the encoder function, which outputs an associated series of continuous forms (z1,..., zn). The decoder function then employs z to create a step-by-step output symbol sequence (y1,..., ym). The model is auto-regressive throughout each phase, which means that it utilises the previously generated symbols as extra input to create the subsequent symbol.[1].

The models we used in our proposed work are: Legal Pegasus [16], Legal LED [17], and DistilBART [18]. They all are based on the transformer architecture and have been pre-trained on large amounts of legal text data.

- Legal pegasus [16] is a modified version of the pre-existing Pegasus CNN DailyMail model from Google. It performs the task of abstractive summarization of legal judgement documents. Allowed maximum input sequence length is: 1024 tokens.
- DistilBART [18] is a seq2seq model which utilizes objectives of a denoising auto-encoder for its training.
- Legal LED [17] has been specifically trained in order to abstractively summarise lengthy legal texts up to 16,384 tokens in length-4.

System architecture of our proposed approach is presented in figure 1. Approach is carried out in four steps: Pre-processing, Approaches for Text Reduction, Integrating Pre-trained models for feature extraction and Text Summarization, and Performance Evaluation of Approaches proposed in this implementation.

To deal with input token size limitation of pre-trained Transformer models and huge size of Judgement documents, we follow the human instinct of abstractive summarization. We reduce the input size by ranking of sentences in document and provide only selected sentences as fine tuned dataset to the Transformer models. Approaches followed for sentence ranking are stated in the algorithms:

Algorithm 2 : Chunking based approach

1: Input the legal document text
2: Pre-process and tokenize the text into a list of words.
3: Define a function to divide the text in smaller chunks and join each chunk to form a string:

```
def text_chunk(txt, size=512):
tok = txt.split()
chunks = [tok[i:i + size]
          for i in range(0, len(tok),
              size)]
return[" ".join(chunk) for chunk in chunks]
```

4: Define a function to generate a summary for each chunk by passing all chunks to the pre-trained seq2seq model:

```
chunks = text_chunk(txt)
for chunk in chunks:
summary = summarize_text(chunk)
final_sumary+=summary
```

5: Concatenate all the chunk summaries to generate final summary
6: Evaluate and Output the final generated summary.

1) Algorithm 1: Sentence Ranking with Keywords
   - Extract keywords that represent the important concepts in the document.
   - Rank the sentences based on the number of keywords they contain.
   - Using the ranked sentences the pre-trained seq2seq model generates the final summary.

2) Algorithm 2: Sentence Ranking using Chunking
   - Break down the document into smaller, manageable sections called as "Chunks".
   - Use a pre-trained seq2seq model to generate a summary for each chunk.
   - Concatenate the summaries for each chunk to form the final summary.
3) Algorithm 3: Sentence Ranking using POS Tagging
   - Identify the Noun and Verbs in text using POS tagging.
   - Rank the sentences based on the occurrences of Noun and Verbs.
   - Using the ranked sentences the pre-trained seq2seq model generates the final summary.
4) Algorithm4: Sentence Ranking using Position of Sentence
   - Slice the text to get the first 15% sentences from the start and last 35% sentences at the end of the judgement document.
   - Using the set of selected sentences the pre-trained seq2seq model generates the final summary.
5) Algorithm 5: Sentence Ranking Sentence Similarity
   - Computes sentence embeddings for each sentence.
   - Compute Cosine similarity for each sentence with every other sentence in the document. If the similarity exceeds the threshold, a "count" is incremented. This count for a senetnce reflects the number of similar sentences.
   - Select the top 40% of sentences with the highest counts.
   - Using these selected sentences the pre-trained seq2seq model generates the final summary.

V. Evaluation metrics

For assessing the effectiveness of a legal document summarization system we need to carefully choose the Evaluation Metric. In this study, we have used several commonly used evaluation metrics to measure the performance of proposed summarization model.

The ROUGE score, which gauges similarity between generated with reference summaries by identifying overlapping n-grams, is the first metric we used. In summarising tasks, the ROUGE score is frequently used to gauge a summarization model's effectiveness. The ROUGE-N score between the system generated summary and the reference summary is:

$$ROUGE_N(\text{gen}, \text{ref}) = \frac{\sum_{r_i \in \text{ref}} \sum_{n\text{-gram} \in r_i} Count(n\text{-gram}, \text{gen})}{\sum_{r_i \in \text{ref}} numNgrams(r_i)} \tag{1}$$

For sets of multiple reference summaries, the ROUGE-N metric is given by:

$$ROUGE - N(\text{gen}, \text{ref}) = \max_k ROUGE_N(\text{gen}, \text{ref}_k). \tag{2}$$

Algorithm 3 : POS Tagging Based Approach

1: Input the legal document text.
2: Pre-process the input text to extract part of speech information and tokenize the words:

```
def txt_postag(txt):
    words = nltk.word_tokenize(txt)
    postags = nltk.pos_tag(words)
    return postags
```

3: Define a function to add Parts of Speech:

```
def add_pos_tags_to_input
(input_txt, pos_tags):
pos_tag_map = {"NN": "<NN>","JJ":"<JJ>"}
  pos_tagged_text = " "
  for word, pos in zip(
    nltk.word_tokenize(input_txt),
    postags):
    if pos in pos_tag_map:
       pos_tagged_text += (
       word + " "
      + pos_tag_map[pos] + " ")
   else:
       pos_tagged_text += word + " "
    return pos_tagged_text
```

4: Generate a summary using the pre-trained seq2seq model.
5: Evaluate and Output the generated summary.

Algorithm 4 : Position of Sentences

1: Input the legal document text
2: Preprocess text by converting it to lowercase, removing non-alphanumeric characters, and replacing multiple whitespaces with single space:
3: Define a function to slice the text to get the first 15% from the start and last 35% from the end:

```
def slice_text(text):
    start = math.floor(0.15 * text_length)
    end = math.floor(0.35 * text_length)
    text = text[start:end]
```

4: Tokenize the sliced text:
5: Generate the final summary using the pre-trained seq2seq model.
6: Evaluate and Output the generated summary.

In the above equations, gen stands for generated summary while reference summary is represented as ref .

The second metric we used is the BLEU score which ranges between 0 to 1. BLEU score is computed by comparing n-grams between the system generated and the standard reference summary.

The BLEU score is computed as: first calculate the clipped n-gram counts, truncate the number of n-grams for each n-gram if it is greater than the number of n-grams found in any one reference combined. The function for clipped counts:

$$Count_{clip}(n\text{-gram}) = \min Count(n\text{-gram}), \text{MaxRefCount}(n\text{-gram}) \tag{3}$$

Algorithm 5 :Sentence Similarity

1: Input the legal document text
2: Preprocess text: convert to lowercase remove non-alphanumeric, replace multiple white spaces with single space
3: Tokenize the preprocessed text into sentences
4: Compute sentence embeddings for each sentence
5: Define a similarity threshold (threshold=0.4) to compute the cosine similarity between sentence embeddings
6: Initialize a count for each sentence to 0 (count = [0]) For every sentence "i", select all sentences "j" and calculate the cosine similarity between their embeddings

```
similarity = cosine_similarity
(sentence_embeddings[i],
sentence_embeddings[j].reshape(1, -1))
 if similarity > threshold:
 count += 1
```

7: Select the top 40% sentences with the highest counts:

```
num_sentences=int(0.4*len(sorted_results))
top_sentences=list(sorted_results.keys())
           [:num_sentences]
sorted_top_sentences =
          sorted(top_sentences,
          key=lambda x:sentences.index(x))
```

8: Generate the final summary using these top sentences and load the model.
9: Evaluate and Output the generated summary.

where, the highest n-gram count is MaxRefCount(n-gram) ever listed for that n-gram in an individual source summary,the n-gram counts are represented by $Count(n\text{-gram})$.

The following formula is applied for calculating the updated n-gram precision scores::

$$p_n = \frac{\sum_{n\text{-gram} \in \text{generated}} Count_{clip}(n\text{-gram})}{\sum_{n\text{-gram} \in \text{generated}} Count(n\text{-gram})}, \tag{4}$$

BLEU score is computed using a "vector of n-gram weights w is used":

$$\text{bleuScore} = \text{BP} \cdot \exp\left(\frac{1}{N}\sum_{n=1}^{N} w_n \log \bar{p_n}\right), \tag{5}$$

For adjusted n-gram precisions, the geometric averages are represented by $\bar{p_n}$, N represents longest n-gram length, and the "Brevity Penalty" (BP), which is calculated by:

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{1-\frac{r}{c}}, & \text{if } c \leq r \end{cases} \tag{6}$$

where, c - summary length and r reference summary length with length closest to the generated summary length.

The third metric is Precision.

$$\text{Precision} = \frac{\text{correctly extracted words}}{\text{total words in generated summary}} \tag{7}$$

The fourth metric is Recall.

$$\text{Recall} = \frac{\text{correctly extracted words}}{\text{total words in reference summary}} \tag{8}$$

The fifth metric is F1 Score.

$$F1\ Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{9}$$

Overall, the use of these evaluation metrics can assist in identifying the most effective approach for each of the models and, subsequently, determining which approach generates the most optimal legal document summary. The insights gained from the evaluation metrics allow for a quantitative analysis of the summarization models, aiding legal professionals in selecting the most appropriate tool for their needs.

VI. Results

We evaluated the performance of three pre-trained models, Legal Pegasus, Legal LED, and DistilBart for generating abstractive summaries of legal documents using different approaches, including chunking, sentence ranking, POS tagging, position of sentences, and sentence similarity. The results are summarized in Table I.

In summary, the chunking approach consistently improved the summarization performance of all three models. Legal Pegasus and Legal LED models outperformed the DistilBart model. Legal Pegasus model with chunking approach performed the best among all the models tested with a Rouge-1 score of 0.4175.

The results presented in the table show the performance of different approaches and models used in generating summaries for legal documents. However, it is essestial to remember that the assessment relied on a comparison between the reference summary made by humans and the summary produced by models.

VII. Conclusion

The work examines the effectiveness of abstractive summarization techniques for legal documents. We evaluated three models, Legal Pegasus, Legal LED, and DistilBART, and tested them with different approaches, including original sentence ranking with keywords, chunking, POS tagging, Position of sentences, and sentence similarity. Through our evaluation metrics, such as BLEU, Precision, Recall, F1 Score, and ROUGE, we were able to compare the performance of each model and approach.

The results showed that Legal Pegasus with chunking-based approach generated the best summary with the highest Rouge score and best scores for recall, precision, and F1 Score. Legal LED with sentence ranking and keywords approach had the best BLEU score.

TABLE I
RESULTS

Results of Legal Pegasus							
	R1	**R2**	**RL**	**Recall**	**Preci.**	**F1**	**BLEU**
Original	0.281	0.140	0.265	0.218	0.560	0.306	0.028
SentRank	0.268	0.123	0.242	0.218	0.459	0.287	0.035
Chunking	0.415	0.248	0.397	0.536	0.391	0.441	0.186
POS	0.269	0.125	0.250	0.207	0.541	0.293	0.021
Location	0.266	0.104	0.229	0.200	0.521	0.282	0.023
Similarity	0.253	0.113	0.234	0.231	0.512	0.343	0.023
Results of Legal LED							
	R1	**R2**	**RL**	**Recall**	**Preci.**	**F1**	**BLEU**
Original	0.220	0.098	0.209	0.174	0.492	0.250	0.011
SentRank	0.266	0.128	0.226	0.234	0.378	0.282	0.287
Chunking	0.375	0.216	0.357	0.510	0.362	0.411	0.155
POS	0.260	0.124	0.222	0.228	0.374	0.276	0.044
Location	0.224	0.097	0.207	0.172	0.497	0.249	0.012
Similarity	0.159	0.067	0.144	0.120	0.416	0.179	0.011
Results of DistillBart							
	R1	**R2**	**RL**	**Recall**	**Preci.**	**F1**	**BLEU**
Original	0.221	0.103	0.143	0.155	0.595	0.242	0.004
SentRank	0.200	0.084	0.128	0.139	0.489	0.213	0.004
Chunking	0.310	0.211	0.278	0.200	0.239	0.193	0.146
POS	0.293	0.146	0.273	0.140	0.491	0.213	0.004
Location	0.210	0.091	0.199	0.149	0.557	0.231	0.004
Similarity	0.207	0.092	0.195	0.144	0.529	0.223	0.005

Our study demonstrates the effectiveness of combining various approaches on encoder-decoder transformers available on Huggingface's transformers library for legal document summarization. We believe that our proposed system can be a valuable tool for legal professionals, making it easier for them to quickly understand the main points of a legal document and make informed decisions. Overall, our study provides performance insights and guidelines for legal professionals who need to summarize legal documents quickly and accurately.

In future work, we aim to expand our dataset and use GPT pre-trained models to enhance legal document summarization. Incorporating advanced models such as GPT-3 could lead to better results, given recent language modeling advancements.

REFERENCES

[1] P. Bhattacharya, K. Hiware, S. Rajgaria, N. Pochhi, K. Ghosh, and S. Ghosh, "A comparative study of summarization algorithms applied to legal case judgments," in *ECIR*, 2019.

[2] A. Shukla, P. Bhattacharya, S. Poddar, R. Mukherjee, K. Ghosh, P. Goyal, and S. Ghosh, "Legal case document summarization: Extractive and abstractive methods and their evaluation," in *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2022*, 2022, pp. 1048–1064.

[3] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, R. Barzilay and M. Kan, Eds., 2017, pp. 1073–1083.

[4] Y. Liu and M. Lapata, "Text summarization with pretrained encoders," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds., pp. 3728–3738.

[5] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, "PEGASUS: pre-training with extracted gap-sentences for abstractive summarization," *CoRR*, vol. abs/1912.08777, 2019.

[6] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *CoRR*, vol. abs/1910.13461, 2019.

[7] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," *CoRR*, vol. abs/2004.05150, 2020.

[8] A. Bajaj, P. Dangati, K. Krishna, R. Uppaal, B. Windsor, E. Brenner, D. Dotterrer, R. Das, and A. McCallum, "Long document summarization in a low resource setting using pretrained language models," in *Proceedings of the ACL-IJCNLP 2021 Student Research Workshop, ACL 2021, Online, JUli 5-10, 2021*, 2021, pp. 71–80.

[9] A. Gidiotis and G. Tsoumakas, "A divide-and-conquer approach to the summarization of long documents," *IEEE ACM Trans. Audio Speech Lang. Process.*, vol. 28, pp. 3029–3040, 2020.

[10] D. de Vargas Feijo and V. P. Moreira, "Improving abstractive summarization of legal rulings through textual entailment," *Artificial Intelligence and Law*, pp. 1–23, 2021.

[11] S. Ghosh, M. Dutta, and T. Das, "Indian legal text summarization: A text normalisation-based approach," *CoRR*, vol. abs/2206.06238, 2022.

[12] M. Schraagen, F. Bex, N. Van De Luijtgaarden, and D. Prijs, "Abstractive summarization of dutch court verdicts using sequence-to-sequence models," in *Proceedings of the Natural Legal Language Processing Workshop 2022*, 2022, pp. 76–87.

[13] C. Khatri, G. Singh, and N. Parikh, "Abstractive and extractive text summarization using document context vector and recurrent neural networks," *CoRR*, vol. abs/1807.08000, 2018.

[14] V. Pandya, "Automatic text summarization of legal cases: A hybrid approach," in *5th International Conference on Advances in Computer Science and Information Technology (ACSTY-2019)*. Aircc Publishing Corporation, 2019.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017.

[16] Legal pegasus. @ONLINE. [Online]. Available: https://huggingface.co/nsi319/legal-pegasus

[17] Legal led. @ONLINE. [Online]. Available: https://huggingface.co/nsi319/legal-led-base-16384

[18] Distilbart. @ONLINE. [Online]. Available: https://huggingface.co/sshleifer/distilbart-cnn-12-6