Simplifying Text - Legal Domain

Bommineni Vishwajith Reddy
EE21BTECH11012

***Abstract*—Understanding legal documents can be overwhelming and often requires specific knowledge that many may not have. With the growing volume of these documents, efficiently extracting key information without reading the entire content is increasingly important. This paper presents a straightforward method to simplify and summarize legal texts, making them easier to digest for everyone, regardless of their legal expertise. By leveraging simple algorithms that focus on the core meanings of texts, our system identifies and presents the most significant information in a brief, clear format. This method not only saves time but also makes legal documents more accessible to non-specialists, ensuring that vital information is not missed. The findings show that our approach effectively distills lengthy legal documents into concise summaries without needing readers to navigate through dense legal information.**

I. Introduction

Text simplification is an integral part of the NLP domain. The idea is to simplify a document to make it more understandable and accessible for people at different education and reading levels while retaining the original text's content. It focuses on lexical simplification and syntactical simplification. In short, we can say simplification is a text-to-text problem that closely resembles text generation or paraphrasing problems.

There are two types of text simplification methods based on the source text corpus.

1) **Sentence simplification :** This method focuses on individual sentences, breaking down complex grammar and substituting difficult words with simpler alternatives. This method ensures that each sentence within a text is easier to understand on its own. However, it doesn't necessarily consider the broader context or connectivity between sentences, which can be crucial for maintaining the overall coherence and narrative of the text.
2) **Document Simplification:** This method encompasses an entire document, aiming to not only simplify each sentence but also to enhance the readability and coherence of the text as a whole. This approach often involves reorganizing information, summarizing parts of the content, and ensuring that the simplified version maintains a logical flow. Document-level simplification is particularly important in scenarios where the structure and interrelation of ideas are complex, such as in legal documents.

Legal documents are a prime example of where document-level simplification is necessary. These documents often contain lengthy passages with multiple interconnected ideas and specialized terms, making them challenging for those without a legal background. Simplifying these documents at the document level ensures that key points and obligations are communicated clearly, making legal rights and procedures more transparent and accessible.

The approach we used, integrates simplification and summarization processes, which are often treated separately in other models, into a unified framework for legal texts. While simplification involves reducing the linguistic complexity of the text, summarization involves condensing the content to highlight only the most critical information by retaining important keywords present in the sentence. Combining these tasks, we aim to provide clear, concise summaries of legal documents that maintain their original intent and crucial legal stipulations.

II. Literature Review

A. Text Summarization

The summarization task can be done in two ways: extractive and abstractive.

1) **Extractive Method :** Retaining most important sentences in text. So, the output is a subset of sentences of the original text. BERT-based extractors achieved a very good performance with this method of summarization.
2) **Abstractive Method :** Involves the generation of summarized text, rephrasing the contents of the input text. It is essentially inspired by the use of transformer-based architecture for language generation. T5 [4] achieved a very good performance with this method of summarization. Another abstractive method involves restraining the generation process to give importance to specific topics.

B. Text Simplification

The Simplification module aims to simplify the original lengthy, wordy, term-specific sentences to easily readable sentences, which involve operations like addition, deletion, and splitting of words in the document. This module maps complex language to semantically similar but simple language(Machine Translation task). Some methods introduced for the task include:

- Statistical Machine Translation (SMT),
- Neural Text Simplification,
- Using Reinforcement Learning,
- Memory Augmentation with Neural Networks, etc.

C. Keyword Extraction

Keyword Extraction [8] includes retaining the most important keywords of text in the final output. Keyword extraction focused on the legal domain is an important step of summarization to retain important keywords specific to the legal domain. There is no specific work focused on the legal domain. Hence, usual methods for keyword selection must be used. Usual methods for keyword extraction include:

1) **Statistical techniques** like word frequency, word co-occurrence, TF-IDF, RAKE(Rapid Automatic Keyword Extraction), etc. RAKE uses the idea of word co-occurrence after preprocessing(removing stopwords) to calculate a score for each word, and then we pick the first T words as important keywords. Here, T is a hyperparameter.
2) **Linguistic techniques** which use morphological or syntactic information. This involves classifying words into PoS tags and then picking T words from the PoS tags, giving importance to different PoS tags based on the information they provide. For example, nouns are given more importance since they hold more information compared to other tags.
3) **Graph techniques** like TextRank. Each vertice is represented by distinct tokens in vocabulary edgeses: directed(dependency between words) or undirected(word co-occurrence). In this case, the degree of a vertex is used to measure importance.
4) **ML techniques** like SVM, Neural Networks, etc.

III. Method

We used a novel model mentioned in [1], specifically designed for text simplification at the document level, comprising two primary elements: a **Summarizer** transformer and a **Simplifier** transformer. These components work together in an integrated manner to achieve end-to-end training for the task of document-level simplification. This design is driven by the need to both preserve essential content from the original document, where a summarization component proves valuable and to enhance readability, which is facilitated by the simplification component.

This model's operation begins with a pre-trained Summarizer, followed by the Simplifier that receives the Summarizer's output without any decoding by the tokenizer. This approach allows for seamless end-to-end training. Avoiding re-tokenization after summarization is crucial, as it would hinder the flow of gradients during training. The logic behind first summarizing and then simplifying is based on observations that reversing this order could complicate the language of already simplified text. Additionally, available datasets for text summarization and sentence-level simplification allow for fine-tuning each component on these specific tasks.

Regarding technical infrastructure, this model, SIMSUM [1], initializes using the pre-trained models BART [3] and T5 [4] models, both of which are recognized for their effectiveness across various NLP applications, including text summarization. For one version of SIMSUM, we use BART models pre-trained specifically for summarization for both the summarizing and simplifying stages. Alternatively, in another configuration, we utilize T5 models in a similar dual-stage setup. Both variants are further fine-tuned on the MILDSum dataset.

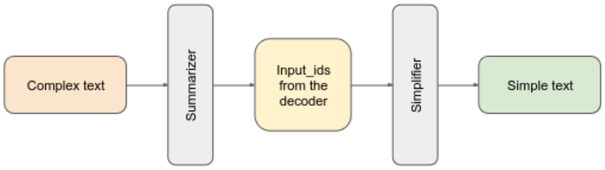

Fig. 1. Worflow of the SIMSUM framework

A. Embedding Similarity

One of the methods for training sequence-to-sequence Transformer models involves using standard maximum likelihood estimations and cross-entropy loss. However, this technique can be enhanced by incorporating an additional loss term as in fig2, which encourages the model to produce text more closely aligned with target outputs. Consequently, we can use a loss function comprising the original cross-entropy loss (L_1) and a term for cosine similarity (L_{CosSim}), defined as follows:

$$L = L_1 + \lambda \cdot L_{\text{CosSim}}$$

$\lambda > 0$ is a hyper-parameter to modulate the influence of the additional term.

Our focus is to augment the similarity between the embeddings of the model's final output and those of the target during training. To achieve this, target embeddings are derived by inputting the target into the Simplifier and using the last hidden state from the encoder for the L_{CosSim} loss term.

Given that we only have access to the summarized text's encoding from the Summarizer, we employ a transformation function $f(\cdot)$ to align the embeddings to the space of simplified text:

$$L_{\text{CosSim}} = -\text{CosSim}(f(H_{\text{sum}}), f(H_{\text{tgt}}))$$

Here, H_{sum} and H_{tgt} represent the embeddings for the summarization and the target, respectively, with dimensions $B \times L \times D_1$, where B, L, and D_1 signify batch size, sequence length, and hidden size, respectively.
The transformation function $f(\cdot)$ is defined as:

$$f(H) = \text{ReLU}(H \cdot W)$$

where W represents a learnable matrix with dimensions $D_1 \times D_2$. Through the application of the ReLU activation function, our approach aims to preserve essential information while filtering out less significant data.

IV. Details of the Base Models used

A. BART

Sequence-to-sequence model pretraining uses the denoising autoencoder known as BART [3], or Bidirectional and Auto-Regressive Transformer. By altering text with a noising function and learning to reconstruct the original content, it is meant

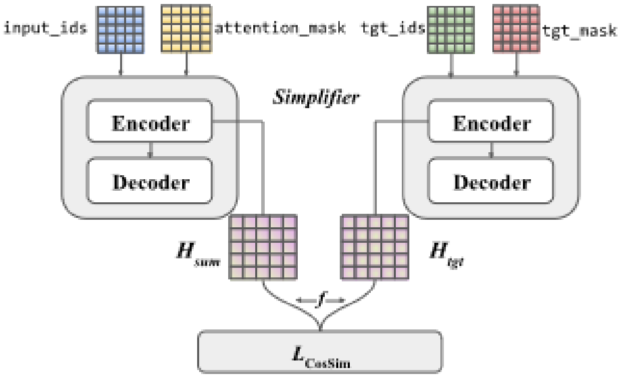

Fig. 2. Embedding similarity

to enhance natural language creation, comprehension, and translation. BART integrates elements from both bidirectional encoders, like BERT, and auto-regressive decoders, similar to GPT, into a unified model structure. This model is particularly adept at text generation tasks but also performs well in comprehension tasks.

BART operates by first corrupting the input text using various noising techniques, including token shuffling and in-filling, where spans of text are replaced with a single mask token. It then trains to reconstruct the original text from this corrupted version, using a sequence-to-sequence framework that employs a Transformer-based architecture. This approach allows BART to handle a wide range of tasks effectively, from translation to summarization, by adapting to the specifics of the input corruption used during training.

Key features of the BART model include:

1) **Denoising Autoencoder:** BART was created to function as a denoising autoencoder, it is taught to get back the original text after a variety of noising techniques have purposefully distorted it. This method aids in the model's learning of a strong grasp of language structure.
2) **Bidirectional and Auto-Regressive Components:** The advantages of auto-regressive decoders (like those in GPT) and bidirectional encoders (like those in BERT) are combined in BART. This enables it to do tasks which require context knowledge from both past and upcoming text tokens in an efficient manner.
3) **Flexible Noising Strategies:** The model supports various methods of corrupting input texts, such as token masking, text in-filling, sentence permutation, and document rotation. These techniques introduce different levels and types of noise, forcing BART to learn more generalizable text representations.
4) **Strong Performance Across Tasks:** BART has demonstrated strong performance on multiple benchmarks, matching or exceeding other models like RoBERTa in comprehension tasks, and setting new state-of-the-art results in text generation tasks such as summarization and dialogue.

B. T5

The T5 model, also known as the Text-to-Text Transfer Transformer [4], is designed to improve upon previous models in natural language processing by using a unified approach. This approach involves treating all language tasks as ”text-to-text” problems, where every task—be it translation, question answering, summarization, or classification—is handled by converting input text into output text using the same model structure and training method.

Key features of the T5 model include:

1) **Unified Framework:** For all text-based activities, T5 has a single model architecture, making it easier to use transfer learning in natural language processing. The use of consistent model architecture and training methods across a variety of professions is made possible by this cohesive approach, which increases efficiency and effectiveness.
2) **Pre-training and Fine-tuning:** The ”Colossal Clean Crawled Corpus,” or C4, is a sizable dataset that is used for pre-training the model. It is a cleansed collection of text that has been taken from websites. The model gains a general knowledge of English from this pre-training, which is subsequently refined for particular tasks to enhance performance.
3) **Transfer Learning:** T5 takes advantage of transfer learning, which is the process of applying pre-training knowledge to improve performance on specific tasks. It has been demonstrated that using this approach greatly improves the model's performance on a number of language processing benchmarks.
4) **Scalability:** The T5 model is designed to scale with increases in dataset size and model complexity. It has been tested in various configurations, from smaller models to very large ones with billions of parameters, achieving top results on many benchmarks.

V. Dataset

The MILDSum dataset [2], which stands for Multilingual Indian Legal Document Summarization, is a vast resource developed to summarize legal documents within the Indian judiciary system. This dataset addresses the significant linguistic challenges in India, where a substantial portion of the population lacks proficiency in English—the primary language used in legal judgments for historical reasons. To mitigate this issue, MILDSum includes 3,122 case judgments from India's higher courts, each summarized in both English and Hindi by legal experts. This dual-language feature is crucial as it not only accommodates linguistic diversity but also enhances the accessibility and comprehension of legal judgments. MILDSum is the first dataset to facilitate cross-lingual summarization, aiming to simplify complex legal language into more understandable summaries, hence allowing fair access to judicial information for speakers of various languages across India.

A. Preprocessing

The dataset was split into 3 sections based on the GFI scores(Above 12, Between 10-12, and less than 10). Among these sections with GFI, The section (with GFI $>$ 12) has more samples, so we used that section to train our model first, and then we experimented with having a training set from all 3 classes. English summary was only considered for ease of training. We think that simplified texts should be shorter than the original documents because they have fewer and easier sentences. Basic preprocessing was done first, which included removing stopwords, irregular spaces, etc. The judgment was divided into about 1200 lines for each file we had, and the same is the case with the summary of the judgment. We appended the entire judgment to one line and the summary to one line. This helped in the better mapping of judgment and summary in complex input to simple input texts, respectively. The preprocessing involves reading text data, possibly cleaning it or transforming it according to defined rules (like tokenization), and then saving the processed data back to the filesystem. The script supports handling different phases of data (like training and validation) and works with different datasets.

VI. Experiments

A. Baselines

We evaluated the SIMSUM methodology as in [1], over the following baselines:

1) T5 model(Text-to-Text Transfer Transformer)
2) BART

We used BART and T5 as they had shown better performances on the text summarization, Hence we used the pretrained versions of them as the initialization of our model. In one model, we used T5 as the backbone for both summarize and simplified. And to adapt it to the legal data, we first fine-tuned the existing T5 model over some part of MILDSum dataset and utilized that model as the simplifier in the SIMSUM architecture. Similarly, BART was used for the both ways of training. We used the HuggingFace to implement T5 model and Pytorch lightning to implement BART. We fine-tuned each model individually of MILDSum dataset. We used the hyperparameters as suggested in the SIMSUM paper [1] as our base.

B. Evaluation metrics

We have used the following metrics based on the following paper [5] on text simplification:

1) SARI - Evaluates text simplification by comparing system output against both the reference and the original text, focusing on how well the system adds, deletes, and keeps words.
2) D-SARI - A modified SARI with penalties for text length, designed for document-level text simplification.
3) FKGL - Measures readability but this metric does not consider the meaning preservation or grammar of the sentence.
4) BLEU - Assesses similarity between simplified text and reference texts by matching sequences of words, with adjustments for text length.

VII. Results

The following table shows the top performance of the models (along with baselines and the simsum) that we trained and explored with diff splits of data.(Better in terms readability of the output generated)

Experiments	**SARI**	**D-SARI**	**FKGL**	**BLEU**
T5	35.23	28.74	6.87	0.86
BART	38.00	30.88	7.18	0.79
SIMSUM(With T5)	42.46	37.36	6.43	0.63
SIMSUM (BART)	40.11	34.36	6.64	0.61

TABLE I
Results on MILDSum Dataset

Here, we can see that in comparison to baseline models, BART is performing better than T5, maybe because of BART's larger context window, and T5 has limitations on the number of tokens passed on the input. And we can see that compared to baseline T5 and BART, SIMSUM with backbone as T5/BART (fine-tuned) performed better in terms of SARI scores. But Bleu's scores are lower for SIMSUM architectures.

We have trained our model for 4 epochs. Even though BART has a larger context window, through these comparisons, we can conclude that, from above, SIMSUM with T5 is performing better than SIMSUM with BART.

Finally, we can conclude that using SIMSUM architecture will increase performance of model compared to baselines but with a lot of computational overhead, since we are having two models.

References

[1] SIMSUM: Document-level Text Simplification via Simultaneous Summarization, Sofia, Xinyu, Martin, 2023

[2] MILDSum: A Novel Benchmark Dataset for Multilingual Summarization of Indian Legal Case Judgments , Debtanu Datta, Shubham Soni, Rajdeep Mukherjee, Saptarshi Ghosh.

[3] BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension Mike Lewis*, Yinhan Liu*, Naman Goyal.

[4] Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, Colin Raffel, Noam Shazeer, Adam Roberts.

[5] Renliang Sun, Hanqi Jin, and Xiaojun Wan. 2021. Document-level text simplification: Dataset, crite- ria and baseline. In Proceedings of the 2021 Confer- ence on Empirical Methods in Natural Language Processing, pages 7997–8013, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

[6] Text Simplification for Legal Domain: Insights and Challenges- Aparna Garimella, Abhilasha Sancheti, Vinay Aggarwal,Ananya Ganesh, Niyati Chhaya1, Nandakishore Kambhatla1

[7] Legal Case Document Summarization: Extractive and Abstractive Methods and their Evaluation

[8] https://monkeylearn.com/keyword-extraction/