Rare Words in Text Summarization

Kunduru Phaneendra Reddy
Department of Information Technlogy
National Institute of Technology Karnataka, Surathkal

Konduru Sai Kiran
Department of Information Technlogy
National Institute of Technology Karnataka, Surathkal

***Abstract*—Text summarization has a significant role in the digesting process of large amount of textual content for faster absorbency. This project discusses using both extractive and abstractive approaches to obtain high quality summaries for content. The extractive phase uses a BERT base uncased Transformer model in conjunction with a pointer-generator layer while the encoder is based on a pre-trained BERT model, and the decoder in an LSTM network. From the PyTorch torch.nn module we utilize learnable embeddings for representation of tokens. F-measure analysis of N-gram frequency guarantees the creation of heterogeneous and substantiated summaries of the lengths of 300 and 600 words. The generated extractive summaries during the abstractive phase are fine-tuned using pretrained T5-small transformer model to provide good quality and concise summaries. For the assessment of the models, the purchased LDC SUMMAC corpus is used through ROUGE assessing the overlapping of the n-gram values and the sentences within the automatically created and reference summaries.**

***Index Terms*—Extractive Summarization, Abstractive Summarization, BERT, T5small, LSTM**

I. Introduction

This paper focuses on summarization, which is a language understanding task within the natural language processing (NLP) field that involves creating a summary of a long document. It has been useful in areas like legal, medical and even in news analysis where large amount of text needs to be read and interpreted fast. The first problem is to preserve the informational communication of the input material and the second is to make the summary short but logically consecutive.

There are two primary approaches to text summarization: extractive and abstractive. Extractive summarization only reconstructs the summary by choosing the most relevant sentences or phrases of the source text, whereas abstractive summarization is to construct the summary by synthesizing the important and coherent information. Although there are such techniques that allow saving the essence of text, they still will have some problems in connection with repetition and absence of proper language usage. While, on the other hand, abstractive procedures can work well providing coherent and accurate summaries but when it comes to facts they seem to be confused or significant details missing.

However, to eliminate these drawbacks, the current work uses both the extractive and abstractive summarization approaches. The extractive phase employs a transformer model based on BERT to perform extraction, but it also has the capability for generation, making it a pointer generator. The final phase utilizes the T5-small transformer model for abstractive adaptation of the extracts, making summaries brief and coherent.

II. Related Work

In recent years solely with the help of deep learning approaches, especially with help of transformer models such as BERT and T5, summarization became outstanding among both extractive and abstractive approaches.

A. BERT and BERTTokenizer

BERT is a Transformer model, which is pre-trained in both directions on a raw text corpus aiming to create deep and bidirectional representations of a text. Proposed by Devlin et al. [1] BERT has been used extensively in the context of natural language processing and especially in the summarization task. BERTTokenizer: The tokenizer that is related to BERT is developed to divide the input string to subword tokens using WordPiece embeddings [2]. This makes the model capable of dealing with new and out of vocabulary words appropriately. The tokenizer also adds the so-called tokeniser special tokens including [CLS] and [SEP] which are useful for classification and sequence labelling.

B. Transformer Model with Pointer Generator Layer

The transformer architecture that Vaswani et al. [3] proposed eliminates the use of recurrent networks and applies self-attention techniques. In order to improve extractive summarization, pointer-generator networks have been incorporated into transformers [4]. Pointer Generator Layer: The layer I am referring to incorporates two mechanisms. Pointer Mechanism: Allows the model to copy the tokens from the input in addition to generating the tokens which can be useful when dealing with out of vocabulary words. Generator Mechanism: Enables the model to predict coins which was not given in the input.

C. Attention Mechanism in T5

The T5 model (Text-to-Text Transfer Transformer) is based on the transformer model and the self-attention mechanism proposed by the authors in the “Attention is All You Need” paper. Standalone attention mechanisms let the model attend to the different parts of input sequential data to identify important relations and patterns for the text processing.

In T5, the attention mechanism is implemented across both the encoder and decoder layers:

Self-Attention in the Encoder: The encoder employs a self-attention to determine dependencies between the words in the input text. Several highlights include; During recurrences helps capture information in context to enable the model to learn long-range word level dependencies and relation ships.

Cross-Attention in the Decoder: The decoder also employs cross-attention to attend to the encoder's output. This mechanism helps the decoder to concentrate on the particular elements of the input sequence that should be covered when the output is produced.

Multi-Head Attention: Applying the multi-head attention function enables T5 to consider numerous features of the input sequences because these sequences are processed within several subspaces at a time. Every head develops different attention patterns, thus improving the comprehending of the textual relationships.

It uses all these types of attention mechanism and in a task such as abstractive summarization, where the model has to produce coherent and realistic summaries from the input text it has to have the ability to direct its attention appropriately.

III. Methodology

A. Data Preprocessing

The first step was to preprocess the dataset to provide clean data for the summarization models selected during the first phase.

Specifically : Where there were data points that did not include a summary, these points were removed from the study.

Some words like is or the were excluded because they don't add any value into the input text, but only increase the noise to the algorithm.

B. Extractive Summarization

In the case of extractive summarization, the BERT Transformer with pointer-generator layer was employed. The architecture consisted of the following components :

Encoder : Instead, they utilized BERT model for feature extraction of contextual information. Decoder: Out of all these RNNs, the LSTM network can produce sequential outputs.

Pointer Layer: An execution layer, First Input-First Output linear layer for maintaining attention and token selection.

Text Embedding: An embedding class which is learnable from the torch.nn module.

The training was for five iterations; input tokens were tokenized using BERT tokenizer, and parameter updates were made using the Adam optimizer.

C. N-gram frequency analysis

A statistical n-gram frequency analysis was applied to the input data to guide the extractive summarization :

The frequency of the N-grams which are the input towards the given model, were pegged at a low probability of inclusion.

Low frequency N-grams in the input data was given high probability of being included.

Based on this probability distribution, the following sentences were produced to form the summaries of sizes 300 words and 600 words for each data sample.

D. Algorithm for extractive approach

The procedure extractive Approach takes in three parameters: dataset, first k, and N. For each input text in the dataset, the following steps are performed: Initialize an empty list called important sentences. Tokenize the input text into sentences using nltk sentence tokenize. Remove stop words from the tokenized sentences. Select the first first k sentences from the processed sentences and store them in important sentences. For each sentence in the tokenized sentences, calculate a score using the function Compute Score For Sentence with a 3-gram approach. Once all sentences have been scored, select the top N important sentences based on the computed scores. Add the selected important sentences to out dataset. The process repeats for each input text in the dataset, and the procedure ends when all texts have been processed.

E. Abstractive summarization

The text samples produced in the extractive stage (300 words and 600 words) were used for the purpose of abstractive summarization with the T5-small model. This model built on those summaries with a filtering and restructuring process that converted the content into a new summary format.

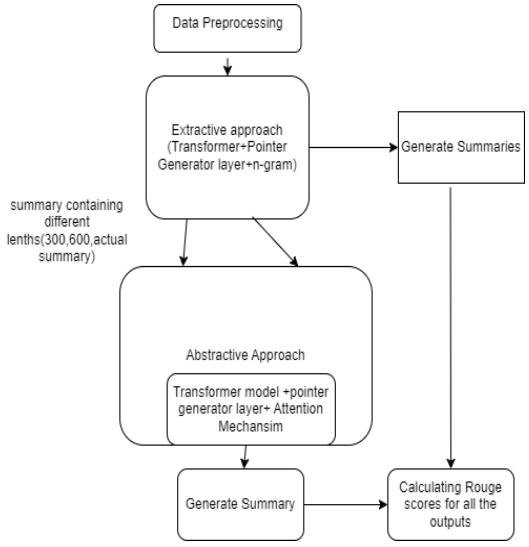

Fig. 1. Example of a figure caption.

Dataset

There is the Indian Legal Dataset available in the Hugging Face with 7130 courts case articles and their summaries. One article covers between 3500 and 5000 words, while the summaries are short and generally do not exceed 300 words each although they can be as short as 150 words. To summarize, this dataset is a rich source of data for the analysis of long-document summarization due to the specifics of legal language and in order to address domain-specific OOV and

frequency issues, which will be presented in detail in the next sections.

I have divided the data in which I have used 90 percent of data for training while staying 10 percent of data for validation data.

Evaluation Metrics

The assessment was conducted with the help of ROUGE (Recall-Oriented Understudy for Gisting Evaluation) scores which are commonly applied for summarization. In their assessments of the radiance of the produced summary from the generated one, ROUGE takes into account n-grams, word sequences and pairs of sentences of the reference summary.

ROUGE-N : Quantifies the intersection between n-grams, which are continuous n words, in the generated and reference summaries.

ROUGE supports three key evaluation metrics:

Precision : Percentage of the given specific pairs out of total pairs in the summary that is generated by the system.

Recall : Share of the equivalent units in the reference summary with the total number of units included in the reference summary.

F1-Score : Considering Precision and Recall equally important, using harmonic mean of Precision and Recall for a balanced result.

Results

F. Rouge scores for all the models

TABLE I

	Rouge-1	**Rouge-2**	**Rouge-L**
Extractive Approach(300)	25.37	7.50	12.31
Extractive Approach(600)	33.26	10.255	14.09
Abstractive Approach(300)	41.71	15.59	25.72
Abstractive Approach(600)	44.58	18.42	27.72

Conclusion

We have successfully implemented the hybrid model using BERT for extractive and T5 for abstractive approach with a transformer and pointer generator layer to archive the decent rouge scores.We investigate how the presence of rare words affects the performance of text summarization, setting aside the extractive approach using BERT and proceeding with the abstractive approach using T5 with a pointer-generator layer. Our findings include the challenge of generating accurate summaries due to rare and domain-specific words, which often contribute to the meaning of the text. Techniques such as augmenting training data with domain-specific vocabulary, also word embeddings, and using more advanced architectures like BERT improved both summary quality and ROUGE scores. Even though there are challenges like limited labeled data available for rare words and coverage vs. conciseness balancing, our results show that advances in NLP, specifically word representations, and preprocessing can boost the robustness of summarization systems to rare words.

References

[1] Devlin, J., Chang, M. W., Lee, K., Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL-HLT.

[2] Wu, Y., Schuster, M., et al. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. arXiv preprint arXiv:1609.08144.

[3] Vaswani, A., Shazeer, N., et al. (2017). Attention Is All You Need. Advances in Neural Information Processing Systems (NeurIPS).

[4] See, A., Liu, P. J., Manning, C. D. (2017). Get To The Point: Summarization with Pointer-Generator Networks. ACL.

[5] Nenkova, A., McKeown, K. (2011). Automatic Summarization. Foundations and Trends® in Information Retrieval.

[6] Raffel, C., et al. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. JMLR.

[7] Hochreiter, S., Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation.

[8] Paszke, A., et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. NeurIPS.