Studying Natural Language Processing for Text Summarization: Proposal

Fraser Wong (301463704), Subhranil Dey (301416515)
CMPT420, Spring 2025. Department of Computing Science, Simon Fraser University
Burnaby, BC, Canada

[1]

Abstract— **Text summarization is critical to Natural Language Processing tasks aimed to minimize the volume of text in large documents into concise, meaningful summaries for fast-paced environments. With the increasing amount of textual data, automated summarization tools are used in everyday applications such as daily news, academic research, and business intelligence. This project will explore transformer-based architectures, such as BART and PEGASUS, to generate high-quality summaries. D depending on the workload and resources, we would like to customize these architectures. The models will be highly tested with modern benchmark datasets such as CNN/DailyMail and XSUM, evaluating performance on PUGUE and BLEU scores. Moreover, the paper will compare transformers to traditional sequence-sequence models, analyze optimizing techniques, and deploy a functional summarization tool.**

I. INTRODUCTION

We believe that with the explosion of online textual content, manual summarization of information has become a tedious, infeasible job for any workload. Automated text summarization offers a solution by condensing documents into digestible formats while preserving key information. Traditional approaches, such as extractive summarization, rely on selecting key sentences to generate summaries. On the other hand, Transformer-based models, particularly pre-trained architectures like BART and PEGUSUS, have demonstrated state-of-the-art performance in summarization tasks by producing human-like fluency.

The project's overall focus is to use transformers to generate abstract summaries of news articles and research papers. Moreover, we want to compare fine-tuned versions of the models to identify the ones that perform reasonably well and based on this information try to pinpoint key characteristics that these models have in common. As a future extension, we would like to tune a model for limited-capability machines and optimize their performance for real-time applications.

II. MOTIVATIONS

We think the need for efficient and accurate text summarization in academics and different sectors of industry is prominent. The tool we are hoping to create will save students and professionals a lot of time in their day increasing their productivity and be creative. News Media, Healthcare, Researchers, Legal, and business documentation, are a few examples just from the top of the head where such a tool would shine. While the existing summarization techniques have improved, challenges such as hallucination, coherence, and factual correctness. These problems can be reduced to a degree but come at a computation cost. When selecting our model, we would consider these issues as we believe that a summarization tool cannot have such biases and stick to the subject material well. In the future, we would like to carry the knowledge gathered from this project to work on architectures fit that could run on Low-power embedded devices.

III. PLANS

A. *Research and Data Gathering (2 weeks)*

We will be learning the theory behind transformers and then reading interesting papers to find further motivations or develop some intuitions about the project, we have found some papers already to guide our studies. Next, we will research the required data to fuel our models, gather it efficiently, and engineer it to fit our needs.

B. *Model Selection and Implementation (2 weeks)*

After finalizing our dataset, we will experiment with different text summarization approaches.

1. Extractive Summarization: Using TF-IDF, PageRank, or transformer-based scoring methods
2. Abstractive Summarization: Implementing pre-trained transformer models such as BART or T5, fine-tuned to our dataset

We will develop our initial implementation in Python using libraries like Hugging Face's Transformers and PyTorch, ensuring modularity for future improvements.

C. *Optimization and debugging (2 weeks)*

Once the base model is implemented, we will focus on improving its performance by:

- Fine-tuning hyperparameters
- Exploring different attention mechanisms and architectures
- Handling edge cases where the model generates incoherent or redundant summaries

[1]

- Implementing techniques such as reinforcement learning to improve summary coherence

Extensive debugging will be conducted using model performance metrics like ROUGE and BLEU scores, as well as qualitative human evaluation

D. *Deployment and Evaluating Results (2 weeks)*

For deployment, we will integrate the trained model into a simple web-based or command-line interface for user interaction. Ideally, a flask-based web app for summarization on demand would provide an optimal user interface.

We will evaluate the final system by:

- Comparing generated summaries with human-written summaries
- Measuring model performance using ROUGE, BLEU, and METEOR scores
- Gathering user feedback on summary readability and relevance.

IV. Related Methods And Strategies

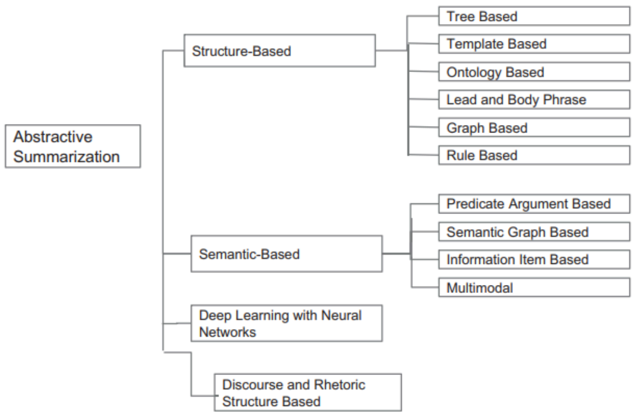
A. *Extraction Summarization*

Extractive summarization generates summaries by selecting key sentences directly from the original text. This method typically relies on word frequency and is one of the simplest approaches to text summarization.

The process begins with text preprocessing, where stopwords, numbers, and punctuation are removed to clean the document. The text is then segmented into individual sentences, which serve as candidates for the summary. To determine sentence importance, word frequencies are calculated by counting the occurrences of each word and normalizing them against the most frequently occurring word. The frequency scores of all words in a sentence are then summed to assign a score to each sentence. Finally, sentences with scores above a predefined threshold are selected to form the summary.

B. *Abstractive Summarization*

Summaries are generated using words that were not in the original text. This method attempts to paraphrase and shorten the length of the original document. This can produce more fluent and concise summaries but requires a deeper understanding of the text's semantics. It also requires more complicated deep learning techniques and sophisticated language modeling. Most abstractive summarization techniques can be classified as either structure-based or semantics-based. Structured-based approaches find the most important information from the text and then use templates to create summaries. They are primarily used along with extractive summary approaches. Here are the different categories of abstractive summarization:



V. REFERENCES

[1] A. P. Widyassari *et al.*, "Review of automatic text summarization techniques & methods," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 4, pp. 1029–1046, May 2020, doi: 10.1016/j.jksuci.2020.05.006.

[2] M. Gambhir and V. Gupta, "Recent automatic text summarization techniques: a survey," *Artificial Intelligence Review*, vol. 47, no. 1, pp. 1–66, Mar. 2016, doi: 10.1007/s10462-016-9475-9.

[3] D. Yadav, J. Desai, and A. K. Yadav, "Automatic Text Summarization Methods: A Comprehensive Review," *arXiv.org*, Mar. 03, 2022. https://arxiv.org/abs/2204.01849

[4] S. Gupta and S. K. Gupta, "Abstractive summarization: An overview of the state of the art," *Expert Systems With Applications*, vol. 121, pp. 49–65, Dec. 2018, doi: 10.1016/j.eswa.2018.12.011.

[5] F. Chiusano, "Two minutes NLP — Four different approaches to Text Summarization," *Medium*, Jan. 22, 2022. [Online]. Available: https://medium.com/nlplanet/two-minutes-nlp-four-different-approaches-to-text-summarization-5a0ce9c06c74#:~:text=Text%20summarization%20approaches,combined%20to%20make%20a%20summary.