Paper-to-Podcast

CS510: Natural Language Processing, Fall 2023
Final Project Report
Mentor: Dr. Ameeta Agrawal

Team(Group-1):
Manisha Yadav (manisha2@pdx.edu)
Deepika Parshvanath Velapure (velapure@pdx.edu)
Vijayalaxmi Pujar (vpujar@pdx.edu)
Harshita Jaiswal (hj7@pdx.edu)

Abstract:
The proliferation of text data in scientific papers has witnessed exponential growth, necessitating the significance of text summarization. Modern recommender systems and text classification models rely heavily on the analysis of vast datasets. The manual creation of accurate and coherent summaries for lengthy scientific articles is a laborious and time-consuming task. Automated summarization, however, proves to be a time and space-efficient solution, particularly beneficial for training machine learning models.

In the realm of scientific paper summarization, two distinct processes are employed: extractive and abstractive. The extractive technique involves selecting pertinent sentences directly from the source text and extracting them for the summary. Conversely, abstractive summarization techniques generate summaries by interpreting the original scientific text, introducing a higher level of complexity due to the need for contextual understanding. Automated summarization plays a crucial role in facilitating efficient data analysis, saving valuable time and resources in the domain of scientific research and literature.

1. Introduction:
While most efforts in summarizing scientific documents focus on creating short summaries, these brief summaries, like abstracts, might not cover all the important details in a scientific text. Generating longer summaries requires deep knowledge in a scientific field, something often seen in researchers' blogs. So, we decided to create a system that would generate a more comprehensive summary for a given research paper.In order to do so, we used blog posts written by researchers in NLP and Machine Learning as reference summaries(target summaries). These blog posts provide more comprehensive insights into scientific articles, acknowledging the need for a thorough understanding of the content and context in scientific work.
In order to capture a glimpse of the paper we followed the approach of summarizing the paper page-wise and generating a more detailed summary. This approach recognizes the need for a deeper understanding and contextualization of scientific content, acknowledging the complexity and intricacies that come with the information presented in scientific documents.

2. Related work:

2.1 Title Generation Task: The title generation task is closely related to text summarization as it involves generating a short text that meaningfully conveys the gist of the input text. The generated title needs to be precise so that the reader can understand what the article is about.
So, this task focuses on generating a short summary for a given input text. In our work, we focus on generating detailed summaries for the given input text by utilizing the important information present in the complete input text or document.

2.2 SumPubMed Dataset: This dataset is useful for generating long summaries for the given input text related to the medical field. So, we wanted a similar dataset that would use the research papers in the NLP and Machine Learning domain, which can be used to train a model to generate detailed summary for a given input research paper.

3. Methodology:

The objective of our project is to generate a comprehensive summary from a given research paper in a PDF format. The approach we followed was to generate a page-wise summary and combine it to generate the overall detailed summary. Being a bidirectional transformer based model, we fine-tuned the T5 model(Baseline model) and compared its performance with the large language models GPT 3.5 and 4 for the summary generation task.

3.1 Dataset:

The dataset requirement for our project was to have a target summary for each page of all the research papers in the training corpus.

- The selected dataset is NLP domain-specific.
- It has extractive and abstractive summaries. We have selected the extractive summaries for creating our dataset.
- The extractive raw summaries consist of 1705 NLP based research papers along with page-wise target summaries of each research paper. The extractive summaries are based on video talks from associated conferences (Lev et al. 2019 TalkSumm) dataset. Summaries can be found under data/extractive/

3.2 Dataset Name:

LongSumm:
https://github.com/guyfe/LongSumm/blob/master/extractive_summaries/talksumm_papers_titles_url.txt

3.3 Dataset Generation Technique:
Available Dataset Format:

- Text file with the title of each paper and the corresponding URL to load the pdf
- Target Summary format:
    - Each line contains:
        - ■sentence index (in original paper)
        - ■sentence score (i.e. duration)
        - ■sentence itself.
    - The fields are tab-separated
    - The order of the sentences is according to their order in the paper.

3.4 Generated Dataset Format:
We filtered the research papers in the dataset and selected only the papers published on the ACL Anthology website. So, the generated final dataset has a list of JSON objects such that each object has the input_text and target_summary for each page of all the downloaded ACL website based research papers.

- Total research papers: 967
- JSON object format :
    - ○ <input_text> page-wise
    - ○ <target_summary> for that page

3.5 Fine-Tuning T5 model:

T5 model features:

- Bidirectional model - left and right contexts
- attention masks - focus on relevant parts of the input text
- It can be fine-tuned to perform specific text processing tasks using the domain specific datasets.

The generated dataset is used to fine-tune the T5 model.

3.6 Large Language Models GPT 3.5 and 4:

Being trained on massive datasets GPT 3.5 and 4 have large-scale knowledge which helps in contextual understanding of the input text for performing the text summarization task. Also, their few-shot and zero-shot capabilities enable it to perform tasks with minimal or no task-specific training data.
So, we generated the summary for the input research paper using the GPT 3.5 and 4 models by following the zero-shot and few-shot prompt approach.

4. Experiments:

In order to perform text summarization tasks we needed models that can consider the context of the input text and accordingly generate meaningful summaries. So we selected below models:

- T5 Model (Baseline Model):
- GPT 3.5/4

T5 Model:

4.1.1 Selection of the T5-model from its available variants:

- T5-small, text summarization resulted in less tokens which could not generate enough meaningful summary
- T5-large resulted in more tokens but higher computational requirements
- Opted T5-base with a 512-token limit per page to generate longer summaries along with less computations.

4.1.2 Implementation of Text summarization task on t5-base model:

- Research paper pdf in converted into single_column text file as input text(from two-column format)
- References page is excluded from input text.
- Pages are divided into chunks and a summary is generated for each page and combined into a final summary as per goal to generate an elaborate summary.
- Human evaluation of summary as to how contextualized the generated summary.
- The T5-base model works well on small chunks to generate a detailed summary.

4.1.3 Fine-tuning the T5-model:

We fine-tuned the T5 model using the T5-base transformer provided by the HuggingFace library. Below are the hyperparameters used for training the model:

- Number of epochs: We experimented by training the model for 2 and 4 epochs. We observed that the training and validation loss was comparatively less for the 4 epoch model. So, we saved the model trained for 4 epochs for further testing.
- Evaluation_strategy: 'steps'
- Eval_steps: 500 steps
- Batch_size: 7

4.1.4 Testing the fine-tuned T5-model:
On testing the fine-tuned T5 model using the test data, showed duplicate sentences in the generated summary. So we have to include below decoding hyperparameters to get more meaningful summary:

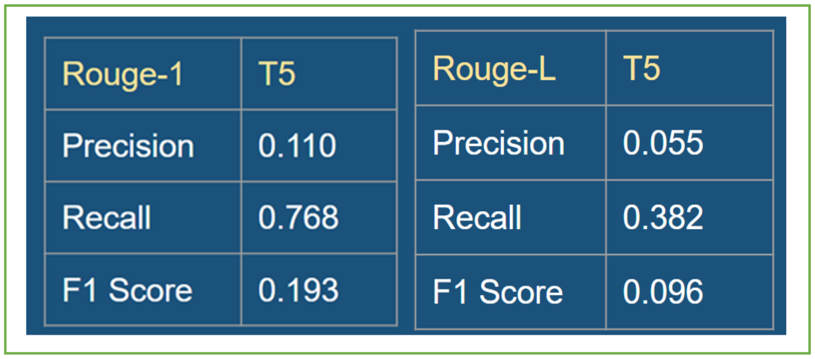
- Temperature: This parameter controls the degree of randomness in the generated summary. We set the value of this parameter to 0.7.
- Num_beans: This parameter helps the model to perform beam search and gather contextual information from the input text. We set the value of this parameter to 5.
- Diversity_penalty: We set the value of this parameter to 1.0 in order to avoid duplicate content and have more diverse text in the summary.
- No_repeat_ngram_size: 3 (This is avoiding the repetition of specified length of n-grams)

4.1.5 Result and Analysis:
We used the reference-free Rouge (Recall-Oriented Understudy for Gisting Evaluation) metric for evaluating the summary generated for a single input research paper. reference free matrix

ROUGE-1 specifically focuses on unigram overlap between the generated summary and the reference summary. It measures the precision and recall of overlapping unigrams (individual words) between the generated summary and the reference summary.

ROUGE-L specifically focuses on the longest common subsequence (LCS) between the generated summary and the reference summary. Unlike ROUGE-1, which assesses unigram overlap, ROUGE-L considers the longest matching sequence of words (subsequence) between the generated summary and the reference summary.

Rouge Score for the summary generated using the fine-tuned T5 model:



Rouge-1	T5
Precision	0.110
Recall	0.768
F1 Score	0.193

Rouge-L	T5
Precision	0.055
Recall	0.382
F1 Score	0.096

Rouge-1 recall value of 0.76 indicated that the summary generated by the fine-tuned T5 model had captured much of the relevant details from the input research paper.

GPT-3.5/4:

4.2.1 Large Language Models(GPT 3.5/4):

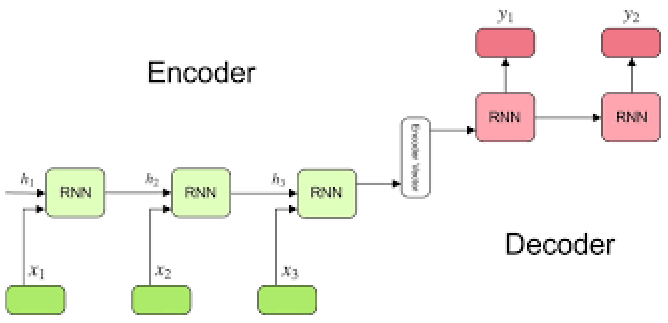
For machines to grasp natural language, they need the capacity to discern connections between words in sentences, and this is precisely where the Transformer concept becomes crucial, especially in models like GPT-3.5/4

These models, driven by the Transformer architecture, is a deep learning model that employs an attention mechanism without relying on traditional recurrent neural networks (RNNs). Diverging from the typical encoder-decoder setups, GPT-3.5/4 employs a singular stack of transformers, eliminating the explicit need for encoder and decoder blocks. This model excels in comprehending context and relationships within textual data, enabling it to produce coherent and contextually relevant responses.

Unlike the described traditional encoder-decoder stack, these models operate as a unidirectional model. It processes input sequences incrementally, generating responses token by token based on learned context. The architecture incorporates multiple attention heads, each focusing on different facets of the input sequence, enabling the model to capture intricate word relationships and generate contextually relevant outputs.

While Google's Transformer model is commonly linked to abstractive text summarization, GPT-3.5/4 surpasses this role with its extensive parameters and capabilities. GPT-3.5/4 not only excels in summarization tasks but also demonstrates remarkable performance across various natural language understanding and generation tasks. Its versatility lies in its aptitude for deducing relationships and context, establishing it as a powerful language model that transcends traditional abstractive summarization to handle diverse language-based applications.

Encoder- Decoder Architecture:



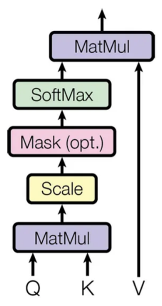
Attention mechanism :

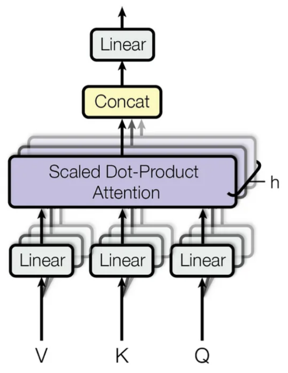
The core of the Transformer architecture lies in the Attention mechanism. This mechanism operates in a sequence-to-sequence manner, where a sequence of vectors is input, and a sequence of vectors is output. Originally designed for machine translation, the Attention mechanism proves versatile and applicable to various tasks involving sequential data. In the context of Attention Is All You Need, the authors define attention as follows:

"An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where a compatibility function of the query with the corresponding key computes the weight assigned to each value."

Multi-head attention is an extension of the attention mechanism used in transformer-based models like GPT (Generative Pre-trained Transformer). In traditional attention mechanisms, a single set of attention weights is calculated for a given input sequence. Multi-head attention introduces the concept of using multiple sets of attention weights, known as attention heads, to capture different aspects of the relationships within the input data.

In summary, the attention mechanism in GPT models allows the model to selectively focus on different parts of the input sequence when generating each word in the output sequence. This enables the model to capture long-range dependencies and contextual information, making it highly effective for a wide range of natural language processing tasks. GPT models, being autoregressive, utilize this attention mechanism across multiple layers to capture intricate relationships and patterns in the input data.





4.2.2 Experiment: GPT 3.5 turbo/GPT 4

Steps outline the comprehensive process of extracting page wise summaries from a research paper using abstract methods, integrating GPT-3.5 and GPT-4 through RapidAPI, and evaluating the performance of the summaries.

Step 1. Extraction of Page Wise Content:

Utilize the PyPDF2 Python library to extract page wise content from the research paper. This involves reading the PDF document and parsing it to obtain text on a page-by-page basis.

Step 2. Preference for Abstractive Summarization:

Acknowledge that abstractive summarization, where new sentences are generated to capture the essence of the entire text, is currently considered superior to extractive methods. Abstractive methods offer summaries that go beyond mere sentence selection from the original text.

Step 3.RapidAPI Integration for Model Inference:

Leverage the RapidAPI host as a quick and efficient solution for integrating APIs into the project. Opt for using pretrained models through APIs for GPT-3.5 and GPT-4 to perform inference on the research paper.

Step 4. API Integration Steps:

1. Sign up on RapidAPI.com to access API functionalities.
2. Identify and finalize the APIs that best suit the project requirements.
3. Generate API keys for the selected APIs.
4. Install and import necessary libraries, including pyPDF2, nltk, requests.

Step 5. Text Processing and Model Integration:

1. Load the research paper text.
2. Parse the text page by page and pass each page's content as input to the API prompt.
3. Utilize the GPT-3.5 and GPT-4 APIs to generate summaries for each page.

Step 6. Summary Compilation:

Concatenate the summaries generated for each page to create a comprehensive final PDF summary of the research paper.

Step 7. Evaluation Using Rouge-Scorer:

1. Import the rouge-scorer library to assess the performance of the summarization.
2. Generate Rouge-1 and Rouge-L scores to evaluate the quality of the summarie produced by both GPT-3.5 and GPT-4.

Step 8. <u>Conversion to Audio Output:</u>

1.Import the gTTS (Google Text-to-Speech) library.

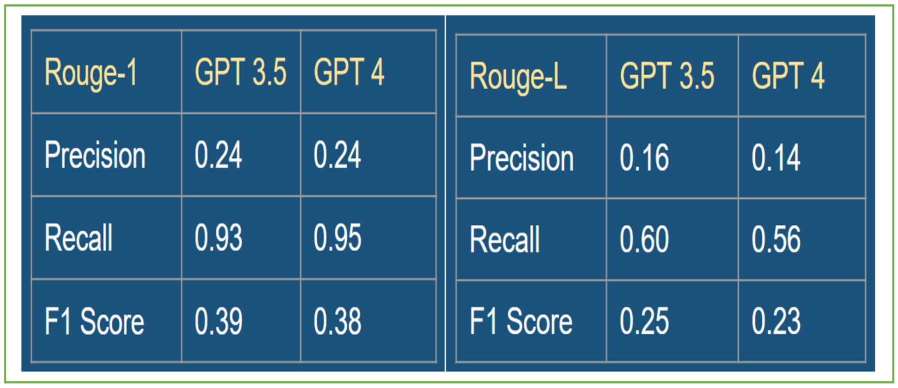
2.Convert the generated summary into an audio file using gTTS, providing an auditory representation of the summarized content.

4.2.3 Results and Analysis

ROUGE-1:
GPT-4 outperforms GPT-3.5, suggesting that GPT-4 produces summaries with a more pronounced match in unigram overlap with reference summaries. This points to an enhanced precision in utilizing individual words or phrases.

ROUGE-L:
GPT-3.5 surpasses GPT-4, indicating that the summaries generated by GPT-3.5 exhibit superior recall for longer sequences or sub-sequences. ROUGE-L, which concentrates on identifying the longest common subsequence, proves beneficial for capturing the comprehensive content and structure of a summary.

Also, a higher temperature setting (e.g., 1.0) introduces greater randomness into the generation process. Summaries generated by GPT-4 with a temperature of 1.0 are likely to display increased diversity, manifesting variations in both wording and structure.

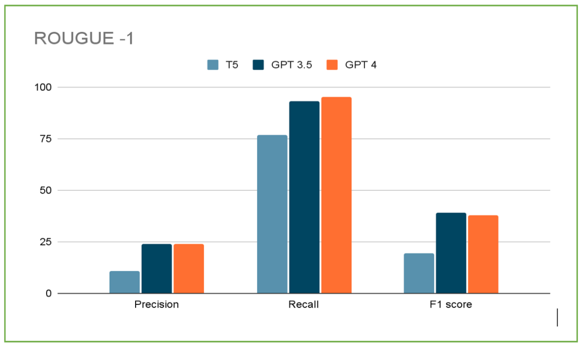


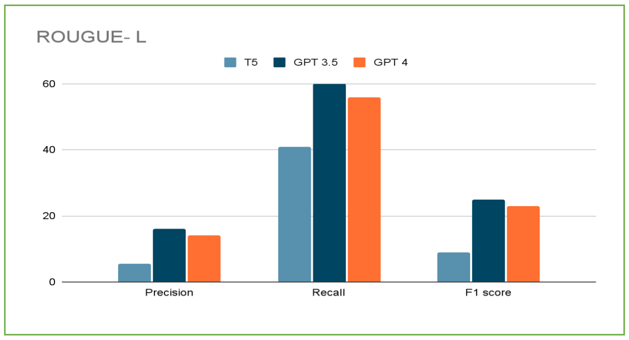
Rouge-1	GPT 3.5	GPT 4
Precision	0.24	0.24
Recall	0.93	0.95
F1 Score	0.39	0.38

Rouge-L	GPT 3.5	GPT 4
Precision	0.16	0.14
Recall	0.60	0.56
F1 Score	0.25	0.23

5 Conclusion:

In converting research papers into podcast episodes, the applied models—T5, GPT-3.5, and GPT-4—show distinctive processing traits. Despite a uniform time requirement of approximately three minutes for conversion, each model exhibits varying approaches to content interpretation. T5 processed the lowest number of tokens at 1098, while GPT-3.5 and GPT-4 handled higher volumes at 1637 and 1677 tokens, respectively. Also, when the summaries generated by all three models are compared with the human written summary for the same test research paper, it is observed that the summary generated by GPT models is more comprehensive and has more relevance with the original research paper. Notably, the resulting audio durations diverged significantly: T5 yielded a concise 6 minutes and 42 seconds, whereas GPT-3.5 extended to 12 minutes and 6 seconds, closely followed by GPT-4 at 11 minutes and 8 seconds. This disparity in audio duration hints at potential differences in content depth and verbosity among the models, raising considerations for optimizing model selection based on desired podcast duration and content richness.





Results(Summary and Podcast):
https://drive.google.com/drive/folders/1-Ed5aANyGTskDUaumfJpko0iJ5IRsWKB?usp=sharing

6. Future Scope

Utilizing tools like Tabula Py and Camelot Python libraries facilitates the extraction of tables and graphs from research papers. With the advent of multimodal capabilities in GPT-4, future applications might encompass extracting content from PDFs containing tables, graphs, and images to generate comprehensive textual summaries. This enhancement would enable a more holistic representation of the data within the papers.

Comprehensive comprehension of image content involves a dual approach: Optical Character Recognition (OCR) for extracting text and employing computer vision techniques for interpreting visual elements. Tools like Pytesseract for OCR and libraries like Pillow (PIL) for image processing facilitate the extraction of textual data embedded within images. Additionally, leveraging computer vision techniques augments the model's ability to interpret and derive meaning from the visual components within research papers.

Processing mathematical equations within research papers requires specialized tools such as MathOCR and Python libraries like Sympy, Mathjax, or Latex for conversion into textual formats. This conversion process allows for mathematical expressions to be translated into a form compatible with NLP models, enabling the incorporation of mathematical concepts into textual summaries effectively.

Generating concise and coherent summaries for multilingual research papers poses a unique challenge. By leveraging the multilingual capabilities of NLP models like GPT-4, the summarization process can encompass a broader spectrum of languages. This involves training the model on diverse multilingual datasets and implementing techniques to ensure accurate and nuanced summarization across various languages, thereby enhancing accessibility and understanding across diverse linguistic backgrounds.

7. Ethical Considerations

Publicly Available Information: The dataset utilized for this project is sourced from the ACL Anthology, containing publicly available research papers. Emphasizing the open-access nature of the dataset ensures clear acknowledgment of the source and respects intellectual property rights.

Transparency in Use: Ensuring transparent communication regarding the use of ACL Anthology data is crucial. It involves explicitly outlining the dataset's utilization in the summarization process. This transparency extends to explaining the methodologies employed and acknowledging any inherent limitations or biases in the dataset or the summarization techniques used.

Computational Resource Responsibility: Mindful utilization of computational resources is essential. This includes employing efficient algorithms and infrastructure to minimize the environmental footprint. Implementing resource-efficient practices not only aligns with ethical considerations but also promotes sustainability in computational processes.

Promoting Academic Integrity: Automated summarization should not replace genuine engagement with research papers. It's crucial to discourage the perception of these summaries as substitutes for in-depth exploration and critical analysis of scholarly works.

Encouraging Original Source Referral: Emphasizing the importance of referring back to the original research papers is paramount. These summaries should serve as aids to comprehension and overview rather than as complete replacements. Encouraging readers to access the full papers promotes a comprehensive understanding, supports academic integrity, and acknowledges the depth and nuances found within the sources.

8. References

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin - " Attention is All you need"

Chandra Khatri, Gyanit Singh, Nish Parikh - "Abstractive and Extractive Text Summarization using Document Context Vector and Recurrent Neural Networks"

Aravindpai Pai - "Comprehensive Guide to Text Summarization using Deep Learning in Python"

Wolf T, Debut L, Sanh V, Chaumond J, Delangue C, Moi A, Cistac P, Rault T, Louf R, Funtowicz M, Brew J (2019) HuggingFace's Transformers: State-of- the-art Natural Language Processing. CoRR abs/1910.03771

https://aclanthology.org/2021.acl-srw.30.pdf

https://github.com/guyfe/LongSumm/tree/master/extractive_summaries

https://paperswithcode.com/paper/classifying-scientific-publications-with-bert

https://www.assemblyai.com/blog/text-summarization-nlp-5-best-apis/#:~:text=What%20is%20Text%20Summarization%20for,into%20their%20most%20important%20parts.

https://www.topcoder.com/thrive/articles/text-summarization-in-nlp

https://deepgram.com/learn/parse-podcasts-with-python-deepgram-asr-text-analysis

https://en.wikipedia.org/wiki/Speech_synthesis

https://huggingface.co/facebook/galactica-6.7b

https://medium.com/nlplanet/a-full-guide-to-finetuning-t5-for-text2text-and-building-a-demo-with-streamlit-c72009631887

https://medium.com/artificialis/t5-for-text-summarization-in-7-lines-of-code-b665c9e40771#:~:text=Developed%20by%20Google%20researchers%2C%20T5,in%20a%20multi%2Dtask%20setting.

https://thepythoncode.com/article/convert-text-to-speech-in-python

https://arxiv.org/pdf/1909.01716.pdf

9 Group contribution:

We agree that all group members made a valuable contribution and therefore believe it is fair that each member receives the same grade for the discussion.