# Research Paper Summarizer

**Afnan Mir**
amm23523
afnanmir@utexas.edu

**Jaykumar Patel**
jnp2369
patel.jay4802@utexas.edu

## Abstract

In this report, we aim to improve accessibility of information presented in complex research papers by using Natural Language Processing (NLP) methods to automatically generate summaries of research papers that are easier to read than the average technical abstract. A dataset of research papers in the medical field which have a technical abstract as well as a plain language summary associated with each paper was used to finetune a pretrained T5 model. The research paper and the technical abstract were used as inputs to the model, and the plain language summary was used as the target output. The finetuned model was evaluated using the ROUGE metric for summarization as well as various readability metrics such as Flesch and Dale Chall. Human evaluation was also used. Our code can be found here.

## 1 Introduction

Research papers are groundbreaking in terms of presenting new findings, theories, and perspectives, making them essential for scientific breakthroughs. However, the knowledge presented in research papers is often inaccessible to the general public for various reasons. Reasons include putting research papers behind paywalls, using dense technical language that the average person cannot understand, and sometimes simply being far too long. This discourages those in the general public who may be interested in the topic of the research paper or those who would like to enter the research area presented in the paper from reading the paper. Though the first issue is not one that can be solved without a change in the current publishing system, the latter two issues can be addressed by using Natural Language Processing (NLP) methods to automatically generate summaries of research papers that are easier to read than the average technical abstract. This would involve finetuning a pretrained model on a dataset to convert the paper and its technical abstract into a plain language summary, which, to some extent, would take down the barrier of technical language and length.

## 2 Dataset and Resources

### 2.1 Dataset

To train and evaluate the model, a dataset of research papers in the biomedical domain was used. This dataset contains 28,124 research papers, each with an associated technical abstract and a plain language summary. The technical abstract is the abstract that is typically found in research papers, but the plain language summary is a summary that is written by the author, where they are required to highlight how the work fits into a broader context in a simple manner without complex acronyms and terminology (Luo et al., 2022). The dataset was split into the train set, the validation set, and the test set, with 26,124, 1,000, and 1,000 papers respectively. The dataset can be found here.

### 2.2 Model Resources

To obtain the resources to load our pretrained model and finetune it, HuggingFace was used, which is an online hub where users and/or companies are able to publish their trained PyTorch or TensorFlow models for others to use. HuggingFace provides a high-level API that allows us to load the model, load the dataset in a proper format, and train the model without worrying about the inner workings of the model.

In order to train the model efficiently, training on a local machine would not suffice. For this purpose, Google Colab was used, where the access of a GPU is provided for free for some limited amount of time.

## 3 Methodology

### 3.1 Model Selection

For the summarization task, a couple of model architectures were researched and explored. The final architecture that was chosen was an encoder-decoder transformer model; namely the T5 (Text-to-Text Transfer Transformer) model (Raffel et al., 2020). This model was pretrained on a mixture of self-supervised and supervised tasks over a large text corpus, which made it a well-performing model for a variety of tasks out of the box. All that had to be done was the prepend the input with the given task (e.g., for summarization: "summarize: ", for translation: "translate English to German: "). The model was chosen because of its ease of use as well as its performance on the general summarization task.

### 3.2 Model Finetuning/Training

To finetune the model, first the data had to be prepared. The initial approach was to use the LongT5 model, which was specialized for tasks with long inputs, and provide as much of each research paper as possible as the input to the model (Guo et al., 2022). However, due to the limited resources provided by Google Colab, it became apparent that using whole research papers as the input to the model would not be possible without running out of GPU memory or truncating our inputs to the point that significant information was lost. Therefore, the approach was changed to use the base T5 model and provide the technical abstract as the input to hopefully create a readable summary.

The data was converted into input tokens that are digestable by the model using the HuggingFace Tokenizer API. The model was then trained using the HuggingFace Trainer API, which allows for easy training of a model on a dataset[1]. The model was trained for 3 epochs, with a learning rate of 2e-5, a maximum input length of 1024 tokens, and a maximum output length of 256 tokens. The training for this would take around 5 hours in the Google Colab environment.

### 3.3 Summary Generation

Once the model was finetuned, it was used to generate summaries for the test set. Initially, the default parameters were used for the generation of the summaries; however, using these parameters led to the generation of summaries that were not very readable for many reasons. One main reason was the constant generation of repeated tokens as seen in the following example:

```
Schistosoma mansoni is one of
the major tropical infectious
diseases responsible for tropical
infectious diseases caused by
Schistosoma mansoni Schistosoma
Mansoni Schistosoma Mansoni
Schistosoma Mansoni Schistosoma
Mansoni Schistosoma Mansoni
Schistosoma Mansoni Schistosoma
Mansoni Schistosoma Mansoni
Schistosoma Mansoni.
```

This case happened quite often in the test set. To mitigate these issues, the following parameters were used for the generation of the summaries:

```
max_length=256, min_length=64,
no_repeat_ngram_size=2,
num_beams=4, early_stopping=True,
length_penalty=2.0.
```

Parameters like `length_penalty` and `early_stopping` prevented the model from generating summaries that were too long because the longer the summaries went, the more susceptible they were to becoming unreadable. The `no_repeat_ngrame_size` parameter was implemented to make sure that the model did not generate the same token over and over again. With the implementation of these parameters, the summaries generated by the model were far more readable and coherent.

## 4 Results and Evaluation

The performance of the model was measured through two primary methods. The first method utilized the comparison of numerical metrics such as the Flesch, Dale Chall, and Rouge scores (Lin, 2004). Essentially, these metrics measure various qualities of summaries, such as unigrams, bigrams, length of sequences, number of syllabus, difficulty of words, etc. However, since the readers of the summaries are ultimately people, using human evaluation is important.

### 4.1 Metrics

After running the various metrics, here are the results of the predicted summaries compared to

[1]Both of the API's can are found in the Transformers library: https://github.com/huggingface/transformers

the original abstracts and original summaries. Although the original text performed better than the predicted text, the predicted text was fairly close in metrics such as Dale Chall, which measures the difficulty of the words in the text, as well as Rouge1 and Rouge2 which specifically looks at the overlap of unigrams and bigrams respectively.

| Metrics Averaged | Predicted Summaries | Original Abstracts | Original Summaries |
|---|---|---|---|
| Flesch | 16.8523 | 27.4492 | 29.3275 |
| Dale Chall | 12.2706 | 11.0360 | 10.8992 |
| Dale Chall v2 | 10.8840 | 9.6439 | 9.7807 |
| Rouge1 | 0.4036 | 0.4987 | 0.4731 |
| Rouge2 | 0.1249 | 0.1785 | 0.1473 |
| RougeL | 0.2140 | 0.4602 | 0.4308 |

### 4.2 Human Evaluation

As aforementioned, the second method for evaluating results was using human evaluation. This involved showing an evaluator a predicted summary and the technical abstract side by side. The specific side that the text was shown on was randomized. Then the evaluator would select which side they thought was easier to read. Here are the results:

| Text | Votes |
|---|---|
| Technical Abstract | 23 |
| Generated Summaries | 20 |

As shown above, approximately 46% humans preferred the generated summaries over the technical abstracts. This is a fairly good result, considering that the model was only trained for 3 epochs, and the model was only finetuned on a small subset of the data. With more training and more data, the model may be able to perform better.

## 5 Future Work

Upon reflection, there were a few evident reasons for the performance of the model. Firstly, the model was only trained on the technical abstracts, which limited the information the predicted summaries was able to capture. Thus, as next steps, the model would be trained on the techncal abstract as well as the introduction and the conclusion of the paper allowing the model to capture more information. Secondly, the model was only trained for 3 epochs, which is not enough to capture the nuances of the data. Thus, the model would be trained for more epochs. Lastly, other metrics would be used to evaluate the model, such as the RNPTC metric.

## References


Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. Longt5: Efficient text-to-text transformer for long sequences.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Zheheng Luo, Qianqian Xie, and Sophia Ananiadou. 2022. Readability controllable biomedical document summarization.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.


## A Group Work Split

For this project, Afnan took responsibility of formatting the data properly to be ingested by the model. He also took responsibility of training the model on Google Colab. Jaykumar took responsibility of generating the summaries for the test set, which also included tuning the parameters of the finetuned model to generate the most coherent model. He also worked on creating the demo for the class presentation. Both Afnan and Jaykumar worked on performing the evaluation of the summaries generated, and they both worked on the presentation and writing the