

# Research Article Summarizer

Team Members: Katyaini Raj, Shivam Soni, Nikhileshwar Reddy Bommareddy, Sai Akhil Rayapudi

## Abstract:

As graduate students, we often have to read a lot of research papers for our studies or out of personal interest. But let's face it, research papers can be super long and sometimes not very exciting. Ever been halfway through a paper and realized it's not what you're looking for? So, how can we figure out the important stuff in less time? Well, that's where text summaries come in. They help us quickly decide if a topic is worth digging deeper into or not. Text summarization is all about turning long texts into shorter, meaningful sentences.

Our primary objective is to use Natural Language Processing, to make text summarization easier. We will be focusing on two key methods: Extractive Summarization, which involves selecting vital sentences from the source text, and Abstractive Summarization, which goes a step further by generating new sentences to capture the essence of the content.

In simple terms, we're working on a way for LLMs to help us get the gist of long research papers faster. This can save us a lot of time and make studying a bit easier. So, we're exploring how LLMs/GenAI can make reading and understanding lengthy texts easier.

Achieving a rouge score greater than the existing general-purpose summarization models when applied to research articles is one of our goals and a Rouge 1 score above 0.4 will be competent with most of the state-of-the-art models out there.

## Methodology:

### 1. Data Collection:

We utilized the Scisumm dataset, a valuable resource within the ACL Anthology network, encompassing more than 1,000 research papers alongside meticulously crafted manual summaries. The dataset, conveniently packaged as a zip file, unfolds into a repository. Unzipping it reveals a thousand distinct folders, each dedicated to a unique research paper and its summary, presenting a substantial and diverse collection for exploration.

Within each of these folders, two key elements contribute to the dataset's richness:

- Document XML:
  - An XML file serves as the backbone of each research paper, containing comprehensive details ranging from the title to the main content. This file encapsulates the essential structure and information of the paper.
- Summary Folder:
  - Here, we encounter a curated manual summary corresponding to the research paper. Crafted by domain experts, these summaries capture the essence of each paper into short paragraphs. The data collection effort, initiated in 2014 by the Yale LILY Lab, represents a collaborative effort by experts in their respective fields.

### 2. Text Preprocessing:

**Step 1 - Finding the Files:** We start by iterating through each of the 1000 folders and identifying two things: the main XML file of the paper and the manual summary in the summary folder.

**Step 2 - Reading and Processing:**

For the XML files, our tool is designed to read through the complex XML format and extract the important parts of the paper, like its title and main content.

For the summaries, we open the summary files and directly load the text file of the summary

Step 3 - Organizing the Data: After gathering all this information, we organize it into a structured format, typically a data frame. Each row in this table represents one research paper, showing the paper's title, its detailed content from the XML, and the corresponding manual summary.

Step 4 – User Input text pre-processing:

As part of our deployment of the model on a website using Flask, after taking input from the user we are performing these pre-processing steps to obtain well-structured summaries:

1. Removing text above the "Abstract" section to eliminate introductory information like title and author names that may not be relevant to the analysis.
2. Removing the acknowledgements section to eliminate non-essential content that doesn't contribute to the main text analysis.
3. Removing the references/bibliography section to exclude citation lists, which are typically not part of the main content.
4. Processing the text with spaCy to remove email addresses, URL's and retaining only alphanumeric tokens to remove special characters and punctuation, focusing on meaningful text elements.
5. Rejoining tokens into a single string to obtain a clean and processed text ready for NLP tasks.

At the end of this process, we get a well-organized, easy-to-use data frame. Each research paper, with its detailed content and summary, is now ready to be used for training our GenAI model. This organized format makes it easier for us to feed the data into the model and ensures that the model learns accurately from the right content and its corresponding summary.

### **3. Tokenization and Train-Test split:**

#### **Pegasus Tokenizer:**

We used the Pegasus tokenizer for tokenization because it is specifically designed for tasks like text summarization and generation. Pegasus is a pre-trained model that excels at understanding the structure of documents and generating coherent summaries. It understands the context and importance of each token within the text, making it a suitable choice for tokenization in the context of text summarization.

In the next phase of our project, we focused on preparing our dataset for the machine learning model. Using Python's sklearn library, we first divided our dataset into two parts: one for training the model and the other for validating its performance. This was accomplished using the `train_test_split` function. We allocated 90% of the data for training and reserved 10% for validation. This split was applied to both the content of the research papers (referred to as content in our DataFrame) and their corresponding summaries (summary).

Once the data was split, the next step involved transforming the text into a format that our machine-learning model could understand. This process is known as tokenization, and it was executed using a tokenizer suitable for our model. The tokenizer processes the text by cutting it into pieces known as 'tokens' and then converting these tokens into numbers that the model can work with. We applied this tokenization to both the content and the summaries of our papers. During this process, we set a maximum length for the tokens (1024 for content and 256 for summaries) to ensure uniformity in data size and added necessary padding to make all data points of equal length.

After tokenizing our data, we wrapped it into a custom dataset class, `SummaryDataset`, which we created using PyTorch, a popular machine-learning library. This class is designed to handle the specific

structure of our tokenized data. Each item in this dataset consists of a set of encodings (which represent the content of a research paper) and labels (which represent the summary). The dataset class also defines how to access individual data points (`__getitem__`) and the total number of items (`__len__`).

Finally, we created two instances of this dataset class: one for training data (`train_dataset`) and another for validation data (`val_dataset`). These datasets are now ready to be fed into our machine-learning model for training and evaluating its performance

#### **4. Model Training:**

##### **Extractive Summarization using Page Rank Algorithm:**

Extractive summarization using the PageRank algorithm is a process where a source document is transformed into a concise summary. It begins by breaking down the document into individual sentences, treating each sentence as a node in a graph. Edges in this graph represent the similarity or importance between sentences, computed using measures like cosine similarity. The PageRank algorithm, known for ranking web pages, is then applied to this sentence graph. It assigns scores to sentences based on their importance within the graph, considering their connections to other sentences. The sentences with the highest PageRank scores are selected and arranged in descending order, forming the extractive summary. This method captures the central and essential sentences in the document, resulting in a coherent and informative summary of the original content.

##### **Pegasus pre-trained model:**

Abstractive summarization using the Pegasus pre-trained model is a process designed to generate concise and coherent summaries of source documents. It starts by taking the input document and encoding it using the pre-trained Pegasus model, which is specifically trained for summarization tasks. Pegasus employs an encoder-decoder architecture, where the encoder reads and understands the document, while the decoder generates a summary. During encoding, the model learns the salient information and relationships within the document. Then, during decoding, it generates a summary by generating new sentences that capture the essence of the content. This abstractive approach allows the model to produce summaries that may not be directly extracted from the input but convey the same key ideas in a more human-like manner. Pegasus excels at this task due to its understanding of document structure and the ability to generate coherent summaries, making it a valuable tool for abstractive summarization tasks.

##### **Pegasus fine-tuned model:**

In our fine-tuning process, we used the Transformers library by Hugging Face to optimize our model. We experimented with over 10 distinct configurations, each involving various hyperparameters and settings. These configurations allowed us to tailor the training process to our specific task. We carefully selected parameters such as the number of training epochs (set at 600), batch sizes for both training and evaluation (both 1), a learning rate of  $2e-5$ , warmup steps (500), and weight decay (0.01) to achieve the desired performance. We have fine-tuned through a trial-and-error approach. Subsequently, we employed the `Trainer` class to carry out training and evaluation on datasets specifically prepared for these tasks.

Upon evaluation, our fine-tuned model exhibited an evaluation loss of approximately 1.38, indicating its proficiency in generating summaries. The evaluation process also yielded valuable insights into the model's performance, including runtime statistics and metrics like samples processed per second and

steps completed per second. This comprehensive approach, characterized by extensive experimentation and meticulous parameter tuning, enabled us to optimize our model for the summarization task at hand.

### **T5 model:**

The T5 (Text-To-Text Transfer Transformer) model represents a formidable transformer architecture renowned for its efficacy in natural language processing tasks, including abstractive summarization. Unlike traditional models, T5 adopts a text-to-text approach, treating summarization as a text-generation challenge. T5 ingests a source document, treating it as a cohesive text entity, and outputs a generated textual summary. Through a pre-training phase focused on denoising autoencoder objectives, T5 hones its ability to reconstruct original text from deliberately masked or altered versions, facilitating an intrinsic understanding of contextual nuances. The model's adaptability is further highlighted in its capability for fine-tuning summarization-specific datasets, where it refines its skills to produce concise and coherent summaries. T5's versatility across a spectrum of natural language processing tasks stems from its innovative approach, positioning it as a potent solution for abstractive summarization and broader language understanding endeavours.

### **5. Output and Evaluation:**

Upon final observation, We have seen a large difference between the fine-tuned and pre-trained Pegasus model, The finetuned model outperformed the pre-trained model making it a go-to choice for research article summarization applications.

Let us consider a climate change paper for example and feed it into 4 models to compare the output produced by each model

#### **Pegasus finetuned output :**

“One of the most noticeable effects of climate change in the past century has been the increase in temperature around the world. The average global temperature has increased by about 1.1 to 1.2 degrees Celsius since 1900. This change has led to a wide range of impacts on the environment, ecosystems, and human societies. One of the primary causes of climate change is the release of greenhouse gases into the Earth's atmosphere. The majority of these emissions come from human activities, including the burning of fossil fuels for energy, deforestation, and industrial processes. One of the most critical impacts is the rise in sea levels caused by the melting of polar ice caps and glaciers, as well as the expansion of seawater as it warms. Shifts in temperature and weather patterns can disrupt the natural habitats of many species, leading to changes in biodiversity. The impacts of climate change extend to human societies as well. For instance, changes in precipitation patterns and temperature can affect crop yields, leading to food shortages and increased prices.”

#### **Pegasus Pre-trained output :**

“One of the most critical impacts is the rise in sea levels caused by the melting of polar ice caps and glaciers, as well as the expansion of seawater as it warms. Shifts in temperature and weather patterns can disrupt the natural habitats of many species, leading to changes in biodiversity. The impacts of climate change extend to human societies as well. For instance, changes in precipitation patterns and temperature can affect crop yields, leading to food shortages and increased prices.”

#### **Extractive using page rank :**

“Climate change refers to significant, long-term changes in the global climate.

One of the most noticeable effects of climate change in the past century has been the increase in temperature around the world.

One of the primary causes of climate change is the release of greenhouse gases into the Earth's atmosphere.

The consequences of climate change are far-reaching and diverse.”

#### **T5 model output :**

“climate change is a complex and urgent issue that impacts the entire planet. climate change is a complex and urgent issue that impacts the entire planet. it demands immediate and sustained action to mitigate its effects and safeguard the future of the environment and human societies .”

The four outputs—Pegasus fine-tuned, Pegasus pre-trained, T5 and the extractive summary using PageRank—offer different perspectives on the topic of climate change. Let's analyze the differences and extract insights:

#### **Pegasus Fine-tuned Output Analysis:**

Provides a comprehensive overview of climate change.

Highlights the increase in global temperature over the past century and attributes it to climate change.

Emphasizes the role of human activities, such as the release of greenhouse gases, in contributing to climate change.

Expands on the impacts of climate change, including effects on the environment, ecosystems, and human societies.

Specifically mentions the rise in sea levels, biodiversity changes, and the impact on crop yields leading to food shortages and increased prices.

#### **Pegasus Pre-trained Output Analysis:**

Focuses on specific impacts of climate change.

Highlights the rise in sea levels due to the melting of polar ice caps and glaciers, along with the expansion of seawater as it warms.

Discusses shifts in temperature and weather patterns affecting natural habitats and biodiversity.

Acknowledges the extension of climate change impacts to human societies, mentioning changes in precipitation patterns, temperature, and their effects on crop yields, leading to food shortages and increased prices.

#### **Extractive Summary using PageRank Analysis:**

Provides concise key points extracted from the text.

Mentions the general definition of climate change and its noticeable effects on global temperature.

Identifies the primary cause of climate change as the release of greenhouse gases.

Acknowledges the diverse and far-reaching consequences of climate change.

#### **T5 Summary Analysis:**

The T5 model succinctly encapsulates the essence of your research article on climate change, stressing its urgent and intricate nature. Emphasizing the global impact of this complex issue, the model underscores the need for immediate and sustained action to mitigate the profound consequences on the environment and human societies. It highlights the primary causes of climate change, including greenhouse gas emissions from human activities, and outlines the far-reaching impacts on ecosystems, rising sea levels, extreme weather events, and threats to food and water supplies. The T5 model advocates for coordinated global efforts, urging measures such as reducing emissions, transitioning to renewable energy, and safeguarding and restoring forests. In essence, it effectively communicates the multifaceted challenges posed by climate change and the imperative for collective action to secure the well-being of our planet and its inhabitants.

In summary, the fine-tuned Pegasus output offers a more comprehensive and detailed exploration of climate change, while the pre-trained Pegasus output focuses on specific impacts. The extractive summary using PageRank provides a condensed version, capturing key aspects of the information.

### Edge Cases:

Case 1: When fine-tuning a dataset with diverse document lengths, the model struggled to handle varying input sizes, causing inconsistent and fragmented summaries.

Case 2: We experimented with different optimization algorithms during fine-tuning, and one Stochastic Gradient descent optimizer resulted in slow convergence and produced summaries with high variance.

Case 3: We fine-tuned the model on a summarization task with limited training examples, and it struggled to generalize effectively, resulting in summaries that lacked depth and coverage.

Case 4: During the fine-tuning process, we encountered difficulties handling extremely long documents, as the model had limitations in processing and summarizing such extensive content.

Case 5: Fine-tuning documents from diverse sources with varying formats and summaries posed a challenge, as the model had difficulty maintaining a consistent style in its generated summaries.

Case 6: Despite experimenting with higher batch sizes during fine-tuning, we found that using a batch size of 1 yielded the best results. Larger batch sizes led to a loss in summarization quality.

Case 7: Extending the number of training epochs proved beneficial, as it resulted in lower loss values and higher ROUGE scores. This demonstrated that allowing the model more time to learn improved its summarization capabilities.

### Rouge Score:

We chose the ROUGE metric (Recall-Oriented Understudy for Gisting Evaluation) for assessing summary quality.

Model	Rogue_1	Rogue_2	Rogue_L
Extractive using Page Rank	0.18	0.1	0.12
T5 transformer	0.36	0.24	0.29
Pegasus Pre Trained	0.4	0.23	0.29
Pegasus Fine Tuned	0.58	0.45	0.5

Table 1. Rouge Score comparison of different models

## Model deployment:

We have deployed this code of our fine-tuned Pegasus model onto a website by generating a .pkl file and loading it using the flask library.

This is how our user input prompt looks like:

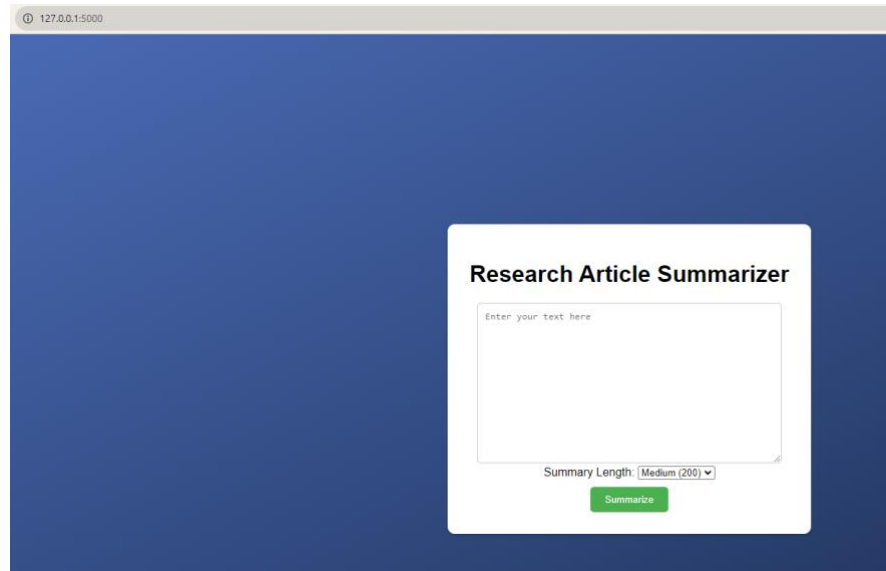
A screenshot of a web browser window showing a web application titled "Research Article Summarizer". The browser's address bar displays "127.0.0.1:5000". The application has a dark blue background. In the center, there is a white rectangular box containing a text input field with the placeholder text "Enter your text here". Below the input field, there is a dropdown menu labeled "Summary Length" with "Medium (200)" selected. At the bottom of the white box is a green button labeled "Summarize".

Fig 1. User input prompt window

Users will have the option to feed in a research article and choose the length of the summary as short – 128 words, medium – 200 words and long – 300 words based on their requirement

The model will perform pre-processing on this data and generate the summary as required and an example output will look like this:

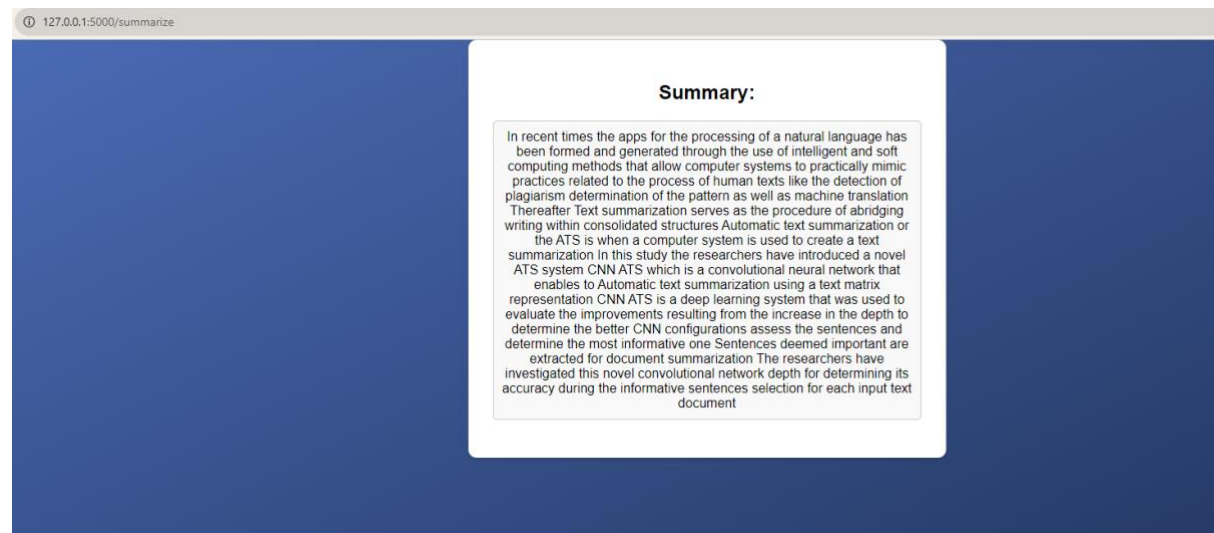
A screenshot of the same web browser window, but now showing the output of the summarization process. The address bar shows "127.0.0.1:5000/summarize". The central white box now displays the word "Summary:" in bold. Below it, a text area contains a paragraph of summarized text. The text discusses the development of natural language processing applications, the use of intelligent and soft computing methods, and the introduction of a novel Automatic Text Summarization (ATS) system based on a Convolutional Neural Network (CNN) for document summarization.

Fig 2. Output generated by model

## Possible Improvements and Considerations:

- Benchmarking: Compare our model's performance with existing summarization models specifically for research articles

- Iterative Training: Continuously improve the model with new data and feedback.
- Ethical Considerations: Be mindful of biases in the data and the potential for misrepresenting research findings.

### **Conclusion:**

Our project is not just about summarizing text; it's about creating a tool that can digest complex research articles and present their essence in a succinct form. The challenges lie in understanding the nuances of the domain, ensuring the summaries are accurate and useful, and dealing with the complexities of language processing.

### **References:**

**Data Source** - [https://cs.stanford.edu/~myasu/projects/scisumm\\_net/](https://cs.stanford.edu/~myasu/projects/scisumm_net/)

*SciSummNet - Scientific Article Summarization Dataset*. (n.d.).

[https://cs.stanford.edu/~myasu/projects/scisumm\\_net/](https://cs.stanford.edu/~myasu/projects/scisumm_net/)

Finetuning Pegasus model - <https://discuss.huggingface.co/t/fine-tuning-pegasus/1433>

**Github project link** - <https://github.com/rayapudisaiakhil/Research-Article-Summarizer>