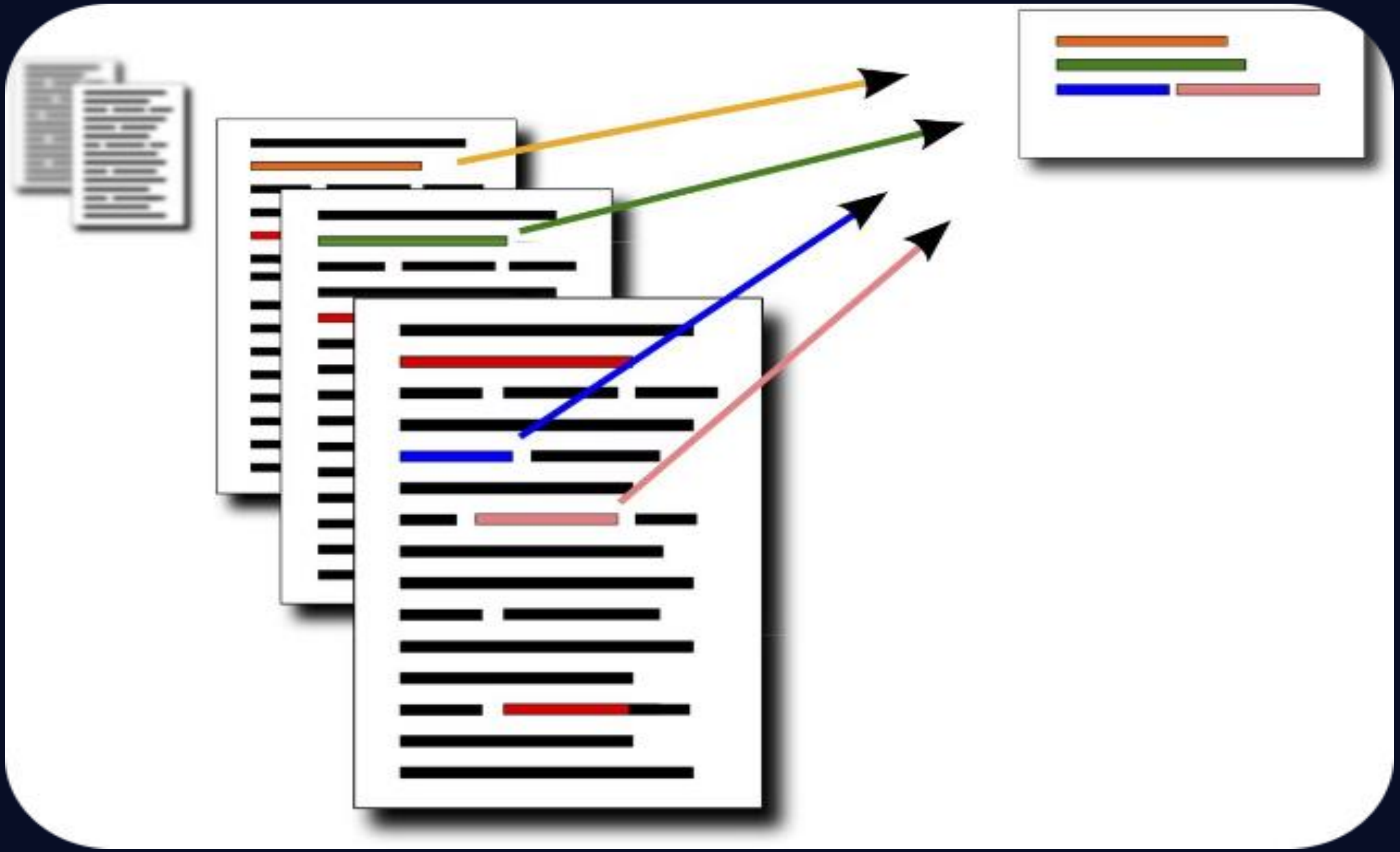
Text Summarizer



Team-1 : Text Summarizer under the guidance of

NARENDRA KUMAR SIR

key points:

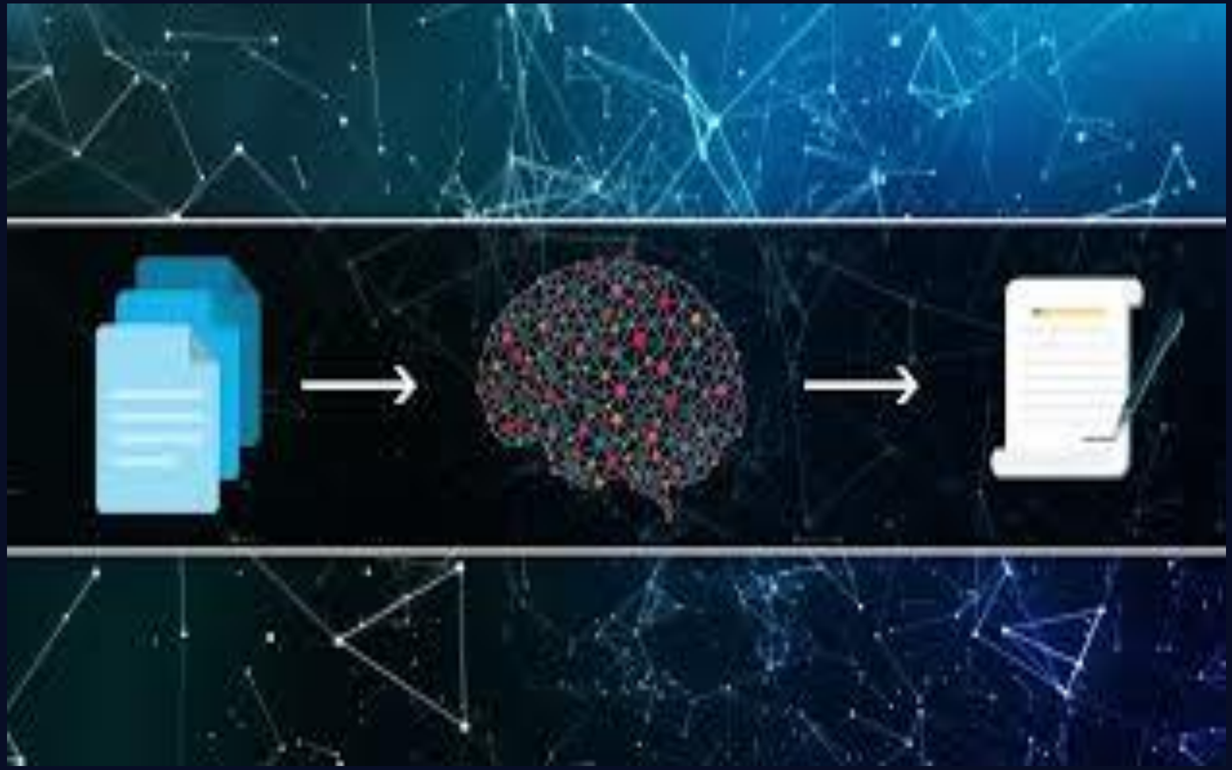
1. Team
2. Introduction
3. Dataset
4. Model
5. Approach
6. Preprocessing
7. Extractive Model
8. Rough Scores-Extractive
9. Abstractive Model
10. Rough Scores-Abstractive
11. Interface-Demo
12. Github Repo

Team-1

- DHARMAVARAPU SANTHI KUMARI
- SAI TEJA ANKAM
- HETU PATEL
- AVANIGADDA SIVAYYA
- VIVEK SADASIVAN

INTRODUCTION

Text summarization is the process of condensing a lengthy text into a concise and informative summary, capturing the key points and main ideas. It is a crucial task in natural language processing with applications in various domains, from news articles to research papers.Text summarization is the creation of a short, accurate, and fluent summary of a longer text document. Automatic text summarization methods are greatly needed to address the ever-growing amount of text data available online. This could help to discover relevant information and to consume relevant information faster.



Dataset

Research:

Considering many dasets and working on them for better understanding we consider two datasets for model.

Xsum.csv

For Abstractive model we choose Xsum.csv datasrt from Hugging face

train.csv

For Extractive model we choose train.csv sataset from Cnn/Daily mails

Overview of T5 Model

1.Transformers

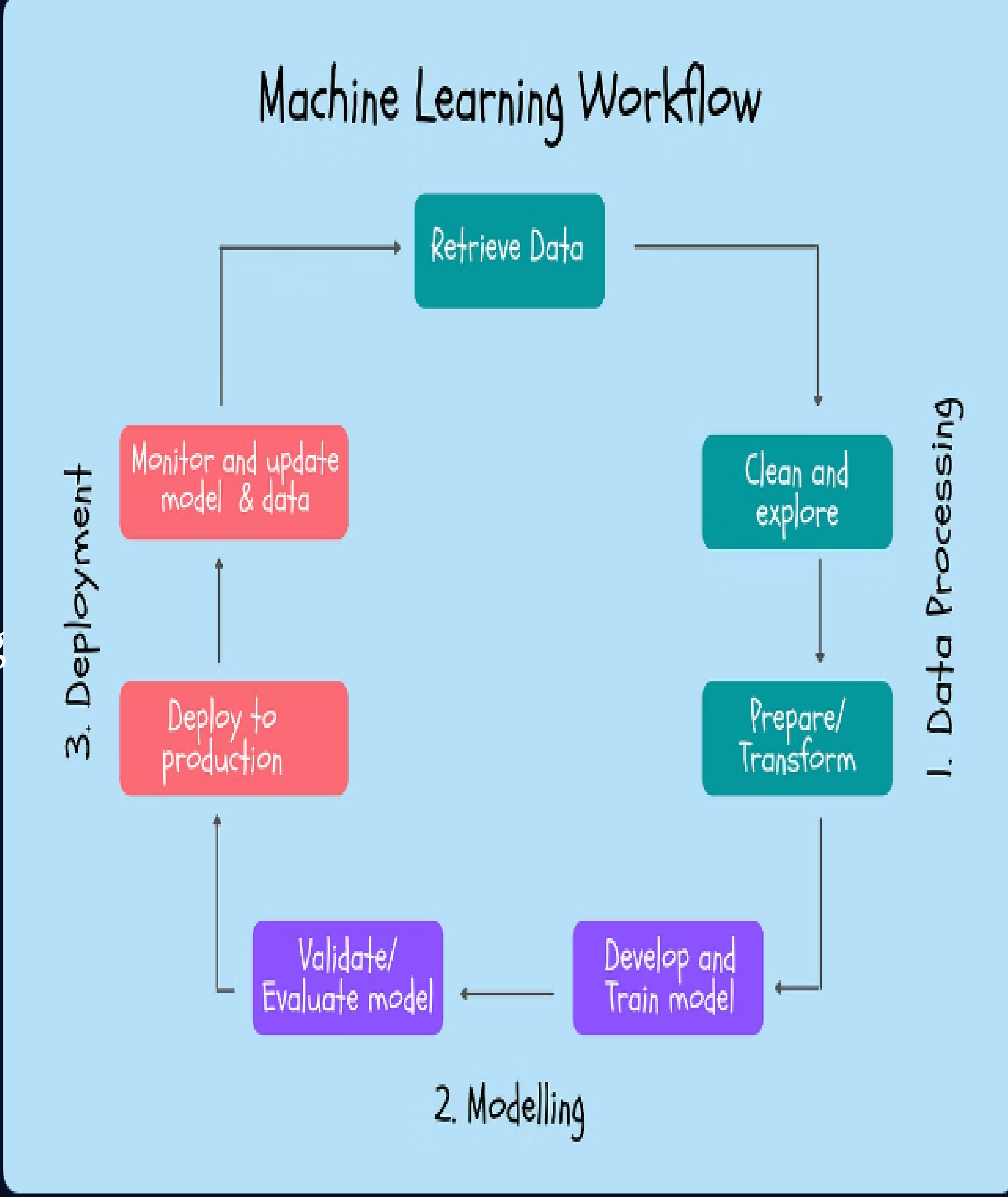
The T5 model is built upon the Transformer architecture,which has revolutionized language understanding and generating tasks.

2.Text-to-Text

T5 is a unified text-to-text model,capable of handling a wide range of natural language processing tasks,including summarization.

3.Pre-training

T5 is pre-trained on a massive corpus of text data,providing it with a strong foundation for fine-tuning on specific tasks.



Approach

1 Extract

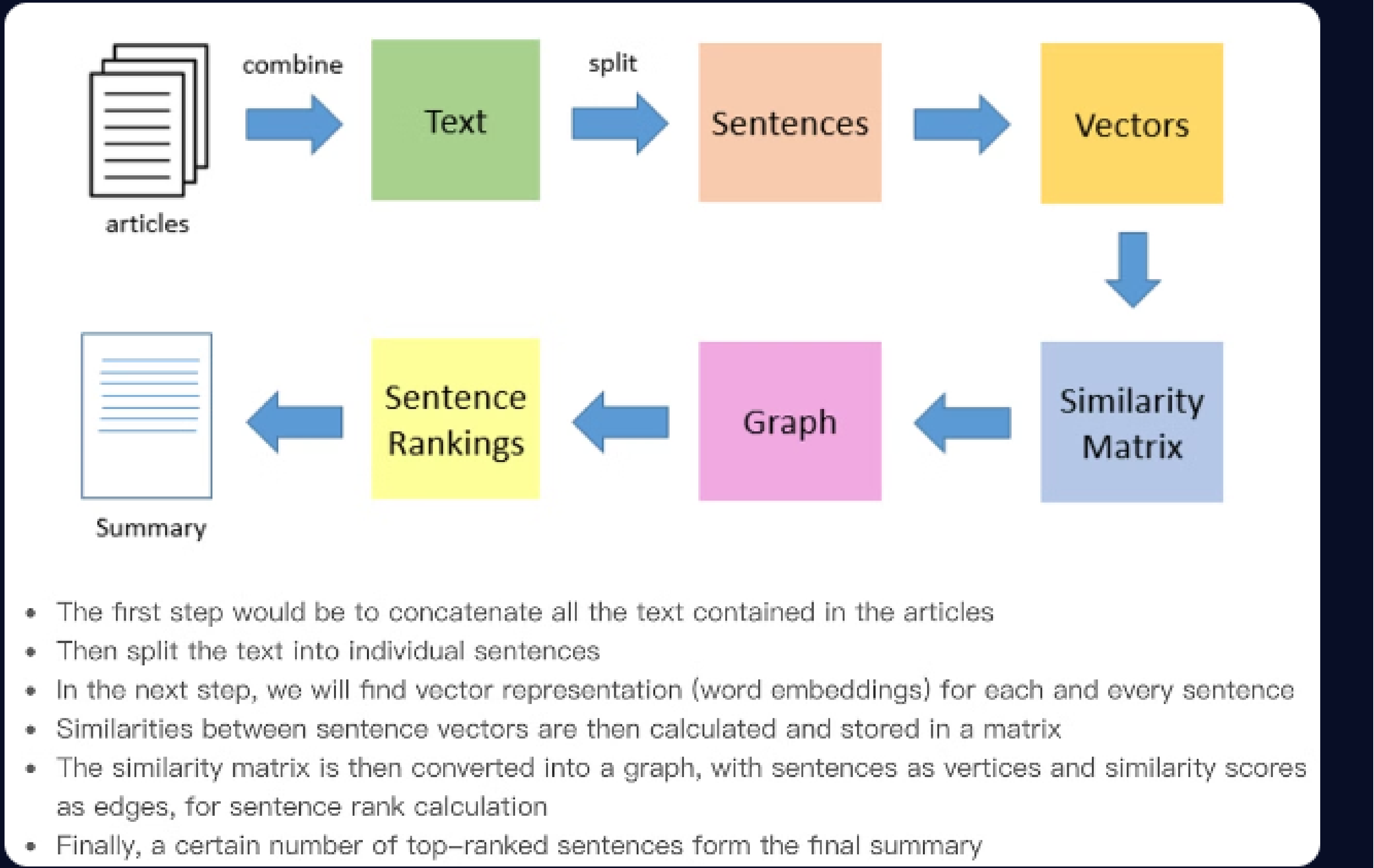
Our model identifies the most essential information in the source text.

2 Compress

The extracted content is intelligently compressed to create a concise summary.

3 Refine

The summary is polished to ensure coherence, clarity, and readability.



- The first step would be to concatenate all the text contained in the articles
- Then split the text into individual sentences
- In the next step, we will find vector representation (word embeddings) for each and every sentence
- Similarities between sentence vectors are then calculated and stored in a matrix
- The similarity matrix is then converted into a graph, with sentences as vertices and similarity scores as edges, for sentence rank calculation
- Finally, a certain number of top-ranked sentences form the final summary

Preprocessing

Cleaning

Removing noise, formatting, and irrelevant content from the source text.

Tokenization

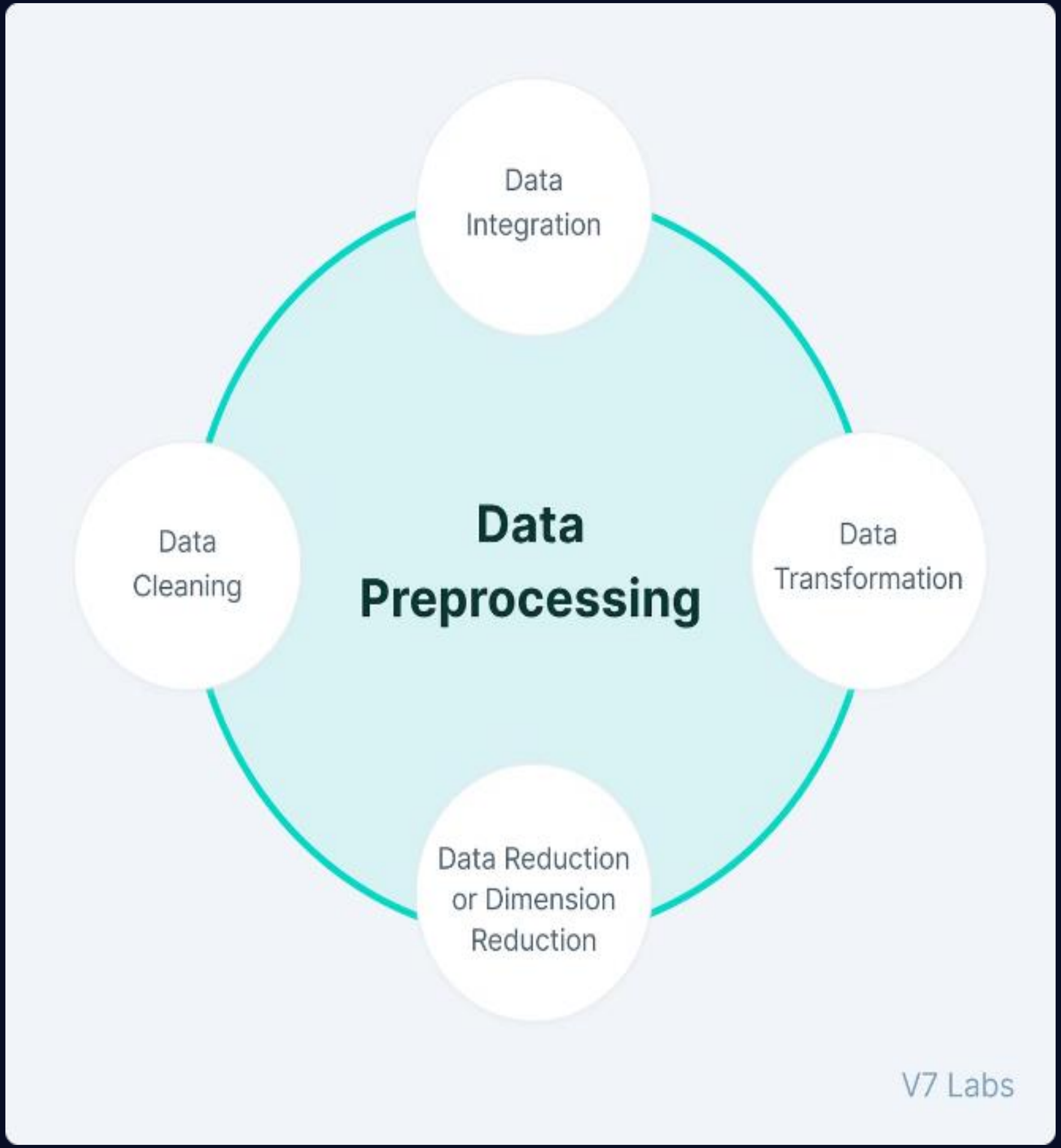
Breaking down the text into individual words, sentences, and paragraphs.

Analysis

Applying natural language processing techniques to understand the text structure and meaning.

Encoding

Converting the preprocessed text into a format suitable for the summarization model.



Extractive Summarization

Key Idea:

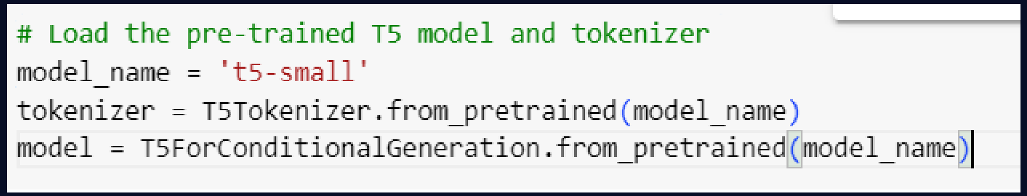
Extractive summarization selects the most important sentences or phrases from the original text and concatenates them to form the summary.

Methods:

"Frequency-based selection.“

Advantages:

Extractive summaries are typically more faithful to the original text and easier to generate.



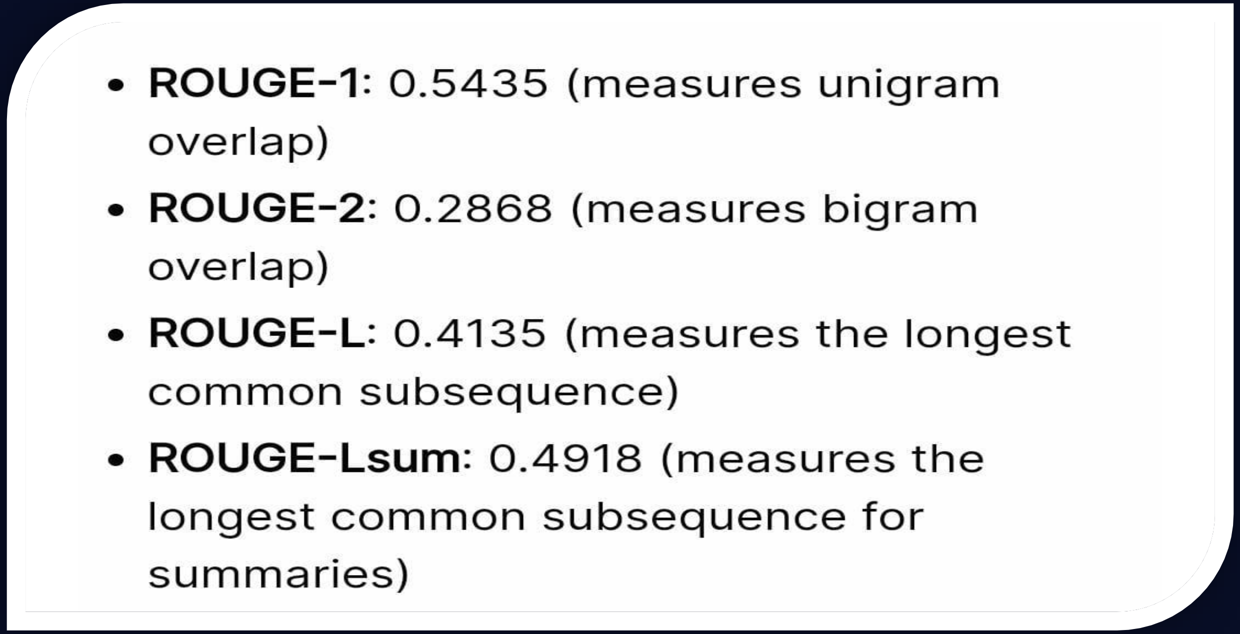
```
# Load the pre-trained T5 model and tokenizer
model_name = 't5-small'
tokenizer = T5Tokenizer.from_pretrained(model_name)
model = T5ForConditionalGeneration.from_pretrained(model_name)
```

Limitations:

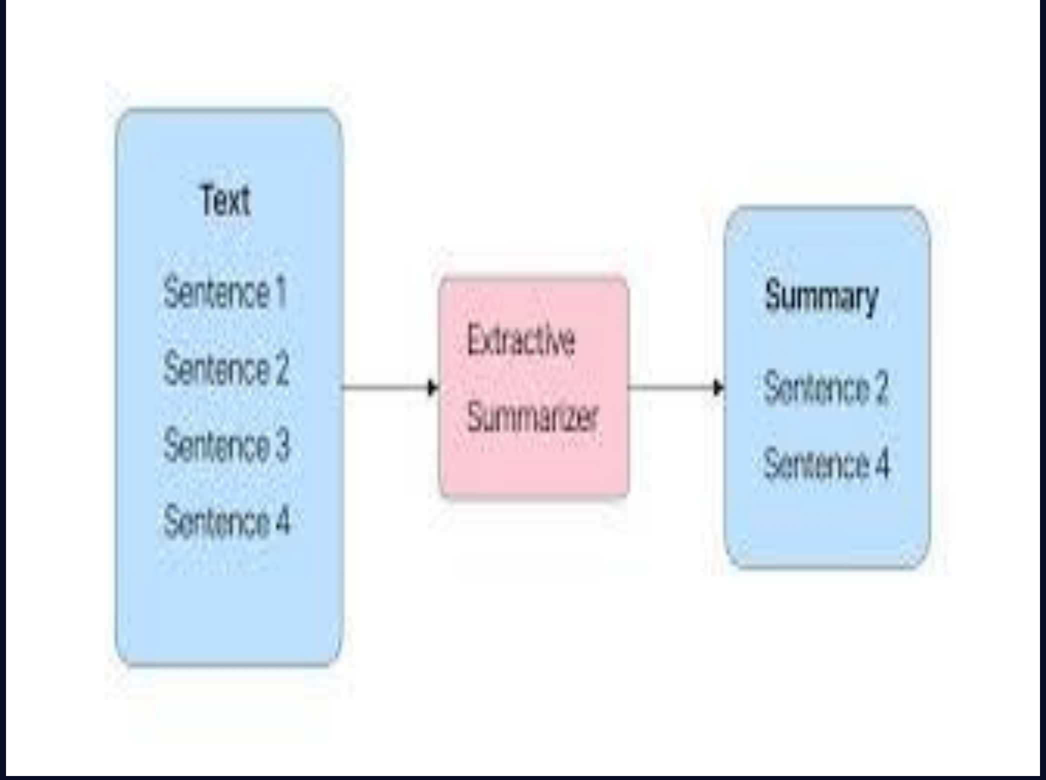
Extractive methods can struggle to capture the overall meaning and coherence of the text, leading To less informative summaries

Rough Scores:

Rough scores for extractive model



- **ROUGE-1**: 0.5435 (measures unigram overlap)
- **ROUGE-2**: 0.2868 (measures bigram overlap)
- **ROUGE-L**: 0.4135 (measures the longest common subsequence)
- **ROUGE-Lsum**: 0.4918 (measures the longest common subsequence for summaries)



Text

Sentence 1

Sentence 2

Sentence 3

Sentence 4

Extractive Summarizer

Summary

Sentence 2

Sentence 4

Abstractive Summarization

Generating New Text: Abstractive Summarization generates new text that captures the main ideas and essence of the original document,rather than just extracting existing sentences

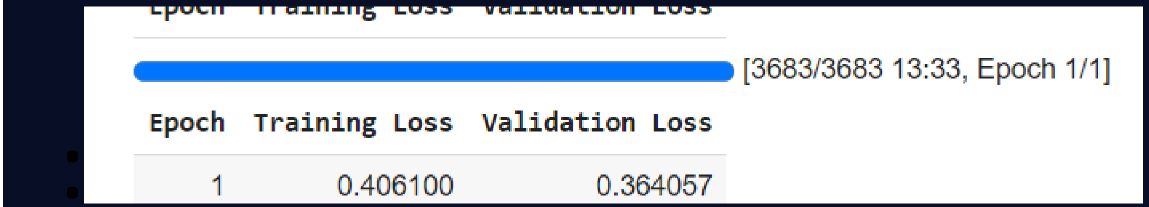
Deeper Understanding: Abstractive models require a deeper understanding of the text to rephrase the content in a concise and coherent manner.

Challenges: Abstractive summarization is more complex and prone to generating inaccurate or irrelevant information if not properly trained.

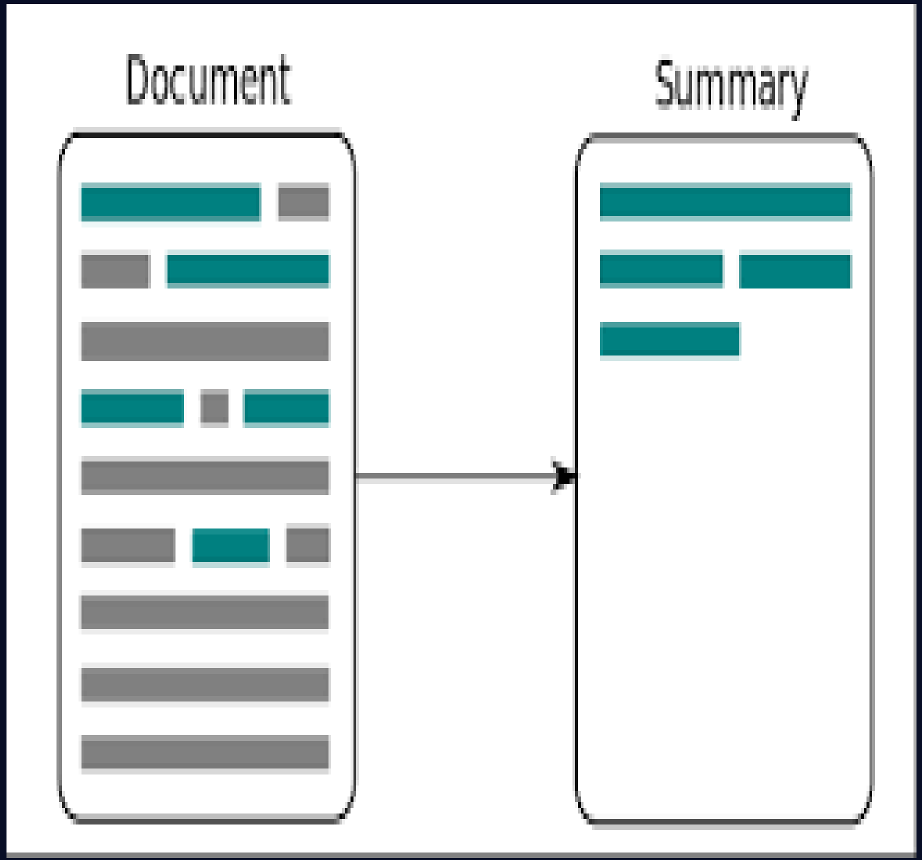
Potential: Abstractive summarization has the potential to produce more informative and human-like summaries that better capture the essence of the original text.

Model training:

```
TrainOutput(global_step=3683, training_loss=0.5255669773256134,
 metrics={'train_runtime': 814.185, 'train_samples_per_second': 18.094,
'train_steps_per_second': 4.524, 'total_flos': 1993855419285504.0,
 'train_loss': 0.5255669773256134, 'epoch': 1.0})
```



```
[3683/3683 13:33, Epoch 1/1]
Epoch  Training Loss  Validation Loss
    1       0.406100         0.364057
```



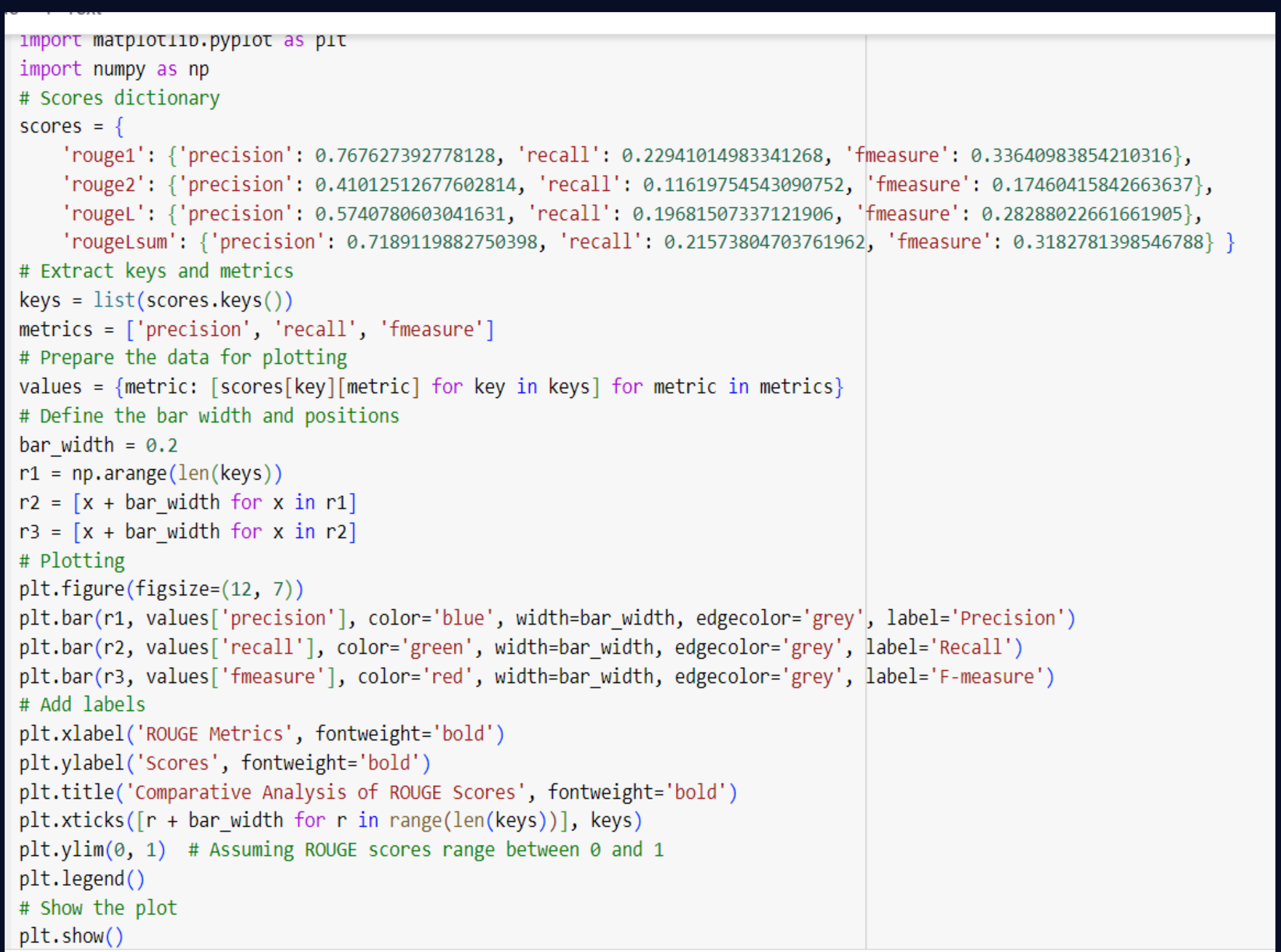
Rough scores

- rouge1: Score(precision=0.767627392778128, recall=0.22941014983341268, fmeasure=0.33640983854210316)
- rouge2: Score(precision=0.41012512677602814, recall=0.11619754543090752, fmeasure=0.17460415842663637)
- rougeL: Score(precision=0.5740780603041631, recall=0.19681507337121906, fmeasure=0.28288022661661905)
- rougeLsum: Score(precision=0.7189119882750398, recall=0.21573804703761962, fmeasure=0.3182781398546788)

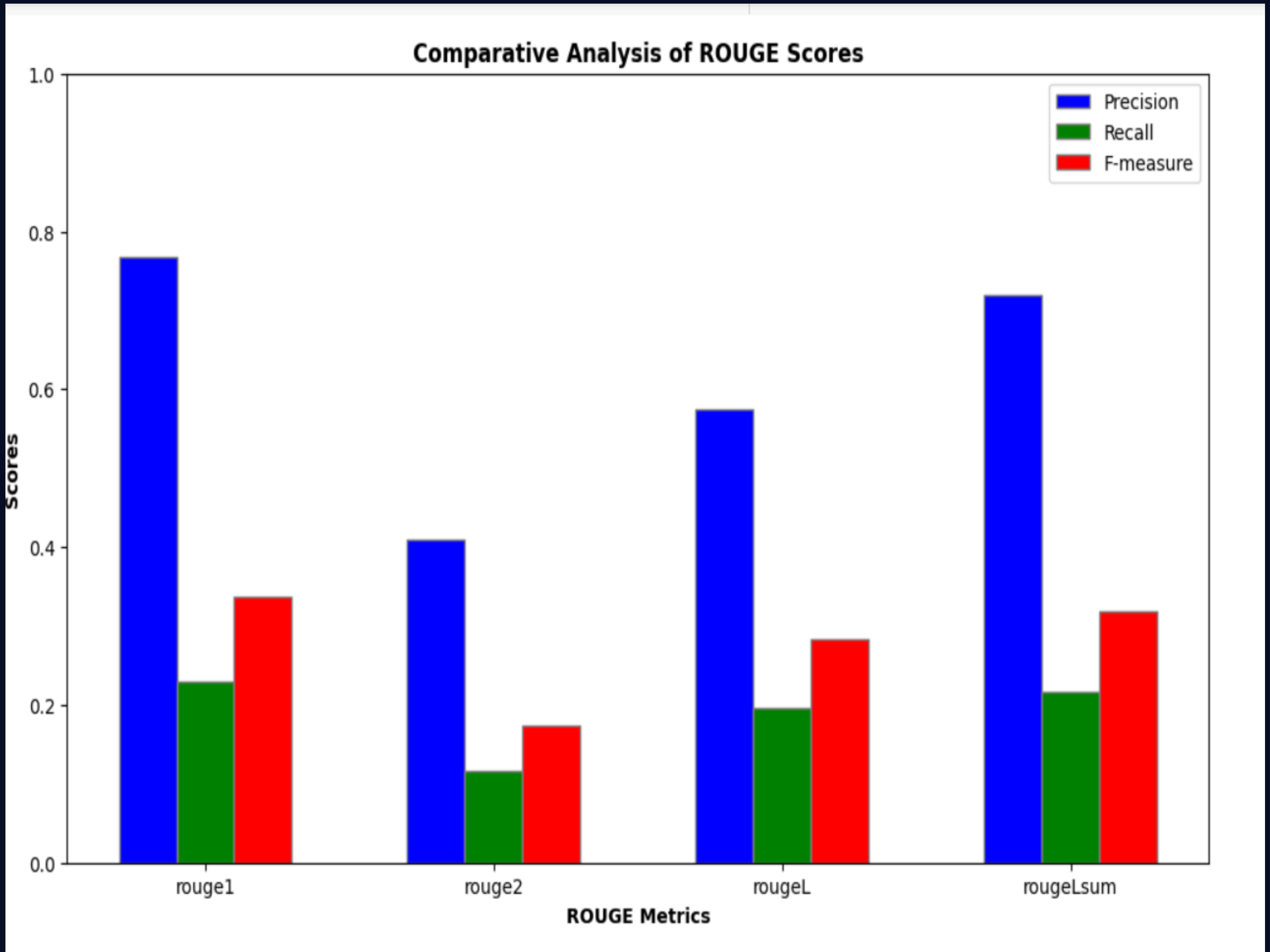
Fine-tuned-Abstractive:
https://drive.google.com/drive/folders/1w2cE6bqU-YomgUloOUafl7zOatt287OY?usp=sharing

For more details check: https://github.com/Patelhlt/text-summarizer/blob/main/Interface.ipynb

Comparative analysis



```
import matplotlib.pyplot as plt
import numpy as np
# Scores dictionary
scores = {
    'rouge1': {'precision': 0.767627392778128, 'recall': 0.22941014983341268, 'fmeasure': 0.33640983854210316},
    'rouge2': {'precision': 0.41012512677602814, 'recall': 0.11619754543090752, 'fmeasure': 0.17460415842663637},
    'rougeL': {'precision': 0.5740780603041631, 'recall': 0.19681507337121906, 'fmeasure': 0.28288022661661905},
    'rougeLsum': {'precision': 0.7189119882750398, 'recall': 0.21573804703761962, 'fmeasure': 0.3182781398546788} }
# Extract keys and metrics
keys = list(scores.keys())
metrics = ['precision', 'recall', 'fmeasure']
# Prepare the data for plotting
values = {metric: [scores[key][metric] for key in keys] for metric in metrics}
# Define the bar width and positions
bar_width = 0.2
r1 = np.arange(len(keys))
r2 = [x + bar_width for x in r1]
r3 = [x + bar_width for x in r2]
# Plotting
plt.figure(figsize=(12, 7))
plt.bar(r1, values['precision'], color='blue', width=bar_width, edgecolor='grey', label='Precision')
plt.bar(r2, values['recall'], color='green', width=bar_width, edgecolor='grey', label='Recall')
plt.bar(r3, values['fmeasure'], color='red', width=bar_width, edgecolor='grey', label='F-measure')
# Add labels
plt.xlabel('ROUGE Metrics', fontweight='bold')
plt.ylabel('Scores', fontweight='bold')
plt.title('Comparative Analysis of ROUGE Scores', fontweight='bold')
plt.xticks([r + bar_width for r in range(len(keys))], keys)
plt.ylim(0, 1)  # Assuming ROUGE scores range between 0 and 1
plt.legend()
# Show the plot
plt.show()
```



c

Interface

User-Friendly Design

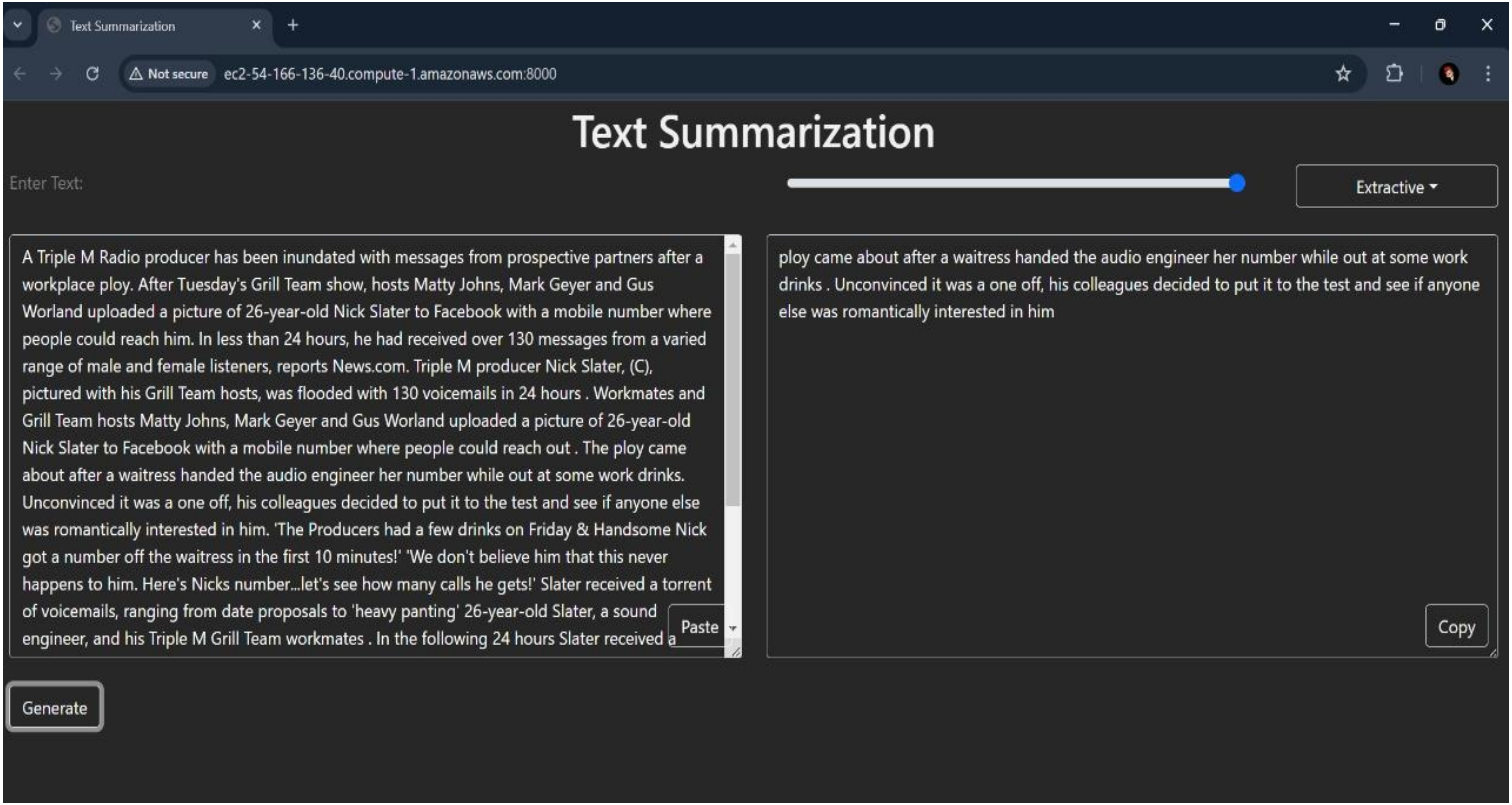
Developing an intuitive and visually appealing interface that makes it easy for users to input text and view the generated summaries.

Interactive Features

Incorporating interactive elements, such as sliders or dropdown menus, to allow users to customize the summarization settings and preferences.

Real-Time Summarization

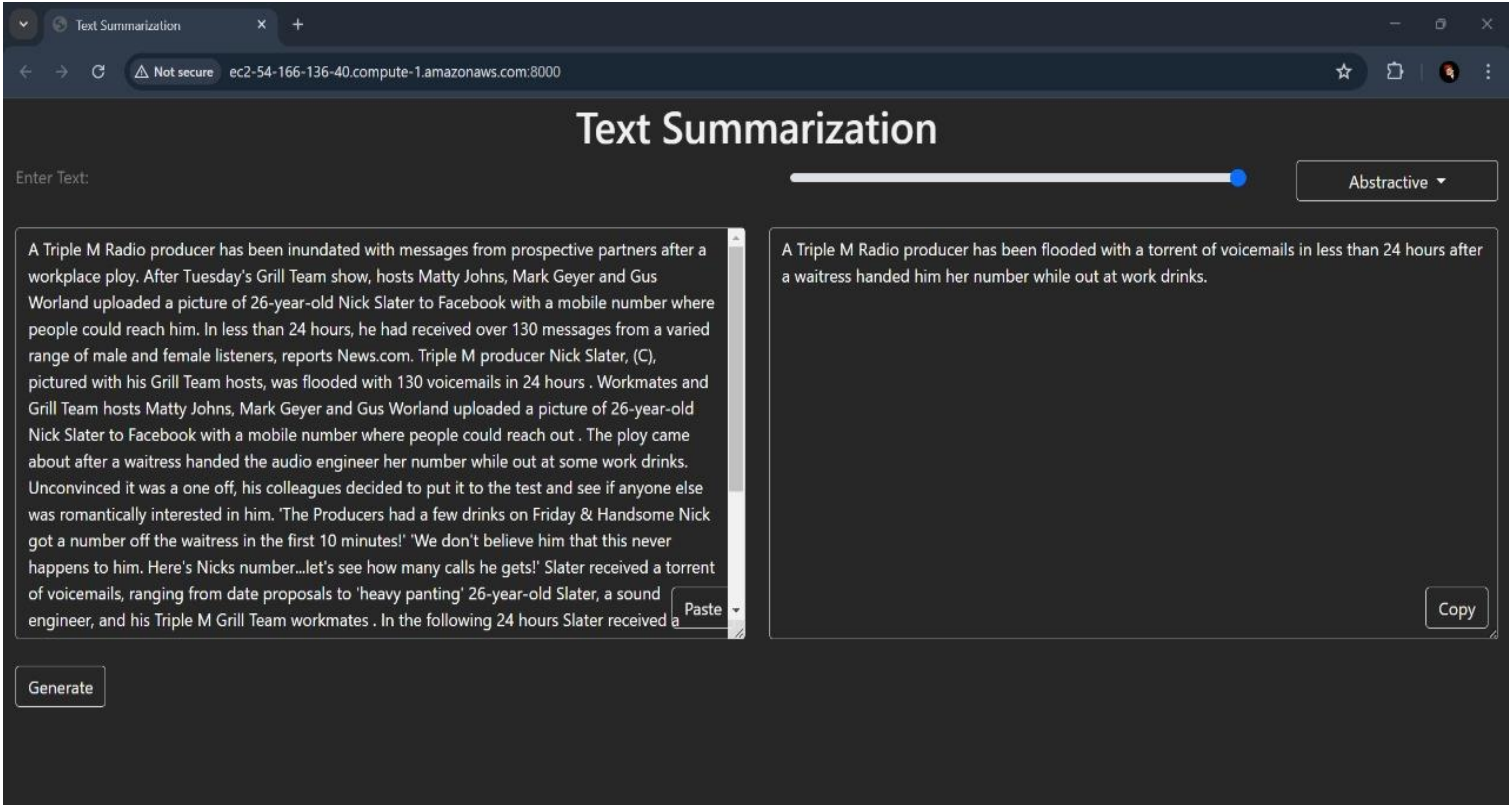
Implementing the text summarization models to provide instant summaries as users type or upload their input text.



Text Summarization

A Triple M Radio producer has been inundated with messages from prospective partners after a workplace ploy. After Tuesday's Grill Team show, hosts Matty Johns, Mark Geyer and Gus Worland uploaded a picture of 26-year-old Nick Slater to Facebook with a mobile number where people could reach him. In less than 24 hours, he had received over 130 messages from a varied range of male and female listeners, reports News.com. Triple M producer Nick Slater, (C), pictured with his Grill Team hosts, was flooded with 130 voicemails in 24 hours . Workmates and Grill Team hosts Matty Johns, Mark Geyer and Gus Worland uploaded a picture of 26-year-old Nick Slater to Facebook with a mobile number where people could reach out . The ploy came about after a waitress handed the audio engineer her number while out at some work drinks. Unconvinced it was a one off, his colleagues decided to put it to the test and see if anyone else was romantically interested in him. 'The Producers had a few drinks on Friday & Handsome Nick got a number off the waitress in the first 10 minutes!' 'We don't believe him that this never happens to him. Here's Nicks number...let's see how many calls he gets!' Slater received a torrent of voicemails, ranging from date proposals to 'heavy panting' 26-year-old Slater, a sound engineer, and his Triple M Grill Team workmates . In the following 24 hours Slater received a

ploy came about after a waitress handed the audio engineer her number while out at some work drinks . Unconvinced it was a one off, his colleagues decided to put it to the test and see if anyone else was romantically interested in him



Text Summarization

Enter Text:

Abstractive

A Triple M Radio producer has been inundated with messages from prospective partners after a workplace ploy. After Tuesday's Grill Team show, hosts Matty Johns, Mark Geyer and Gus Worland uploaded a picture of 26-year-old Nick Slater to Facebook with a mobile number where people could reach him. In less than 24 hours, he had received over 130 messages from a varied range of male and female listeners, reports News.com. Triple M producer Nick Slater, (C), pictured with his Grill Team hosts, was flooded with 130 voicemails in 24 hours . Workmates and Grill Team hosts Matty Johns, Mark Geyer and Gus Worland uploaded a picture of 26-year-old Nick Slater to Facebook with a mobile number where people could reach out . The ploy came about after a waitress handed the audio engineer her number while out at some work drinks. Unconvinced it was a one off, his colleagues decided to put it to the test and see if anyone else was romantically interested in him. 'The Producers had a few drinks on Friday & Handsome Nick got a number off the waitress in the first 10 minutes!' 'We don't believe him that this never happens to him. Here's Nicks number...let's see how many calls he gets!' Slater received a torrent of voicemails, ranging from date proposals to 'heavy panting' 26-year-old Slater, a sound engineer, and his Triple M Grill Team workmates . In the following 24 hours Slater received a

A Triple M Radio producer has been flooded with a torrent of voicemails in less than 24 hours after a waitress handed him her number while out at work drinks.

Conclusion

In conclusion, this project has explored the potential of the T5 model for both extractive and abstractive text summarization.
The results demonstrate the strengths and limitations of each approach, highlighting the importance of tailoring the summarization method to the specific needs of the task and the target audience.

Github Link

https://github.com/saitej-a/Infosys_text_summarizer

Thank you