

TEXT SUMMARIZATION USING T5 TRANSFORMER

Devapangu Abhishek(21bds016), Hosur Sai Kartik(21bds021),
Jarupula Anil Kumar(21bds024), Nischay Kondai(21bds045), Pratik Raj(21bds047)

Abstract—This project explores the effectiveness of the T5 (Text-To-Text Transfer Transformer) model for text summarization. Our study examines the impact of hyperparameters and dataset characteristics on T5's summarization proficiency. The results demonstrate T5's ability to generate coherent and concise summaries, taking into consideration the optimized parameters for enhanced performance. This research contributes valuable insights to the application of transformer-based models in text summarization, providing a detailed guide for researchers.

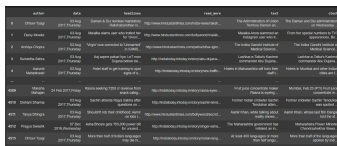
Key Words: Extractive summarization, T5 transformer model, Natural language processing

I. INTRODUCTION

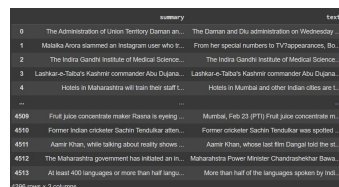
We have a lot of information or stories from around the world that we read in form of text, whether in newspapers, news articles, magazines, books, etc. But, it happens many times that we may want to gain this information within a relatively shorter span of time due to limited availability of time. Hence, we need some mechanism which can reduce the information size by removing the unnecessary or less important contents. One of the solution may be a text summarizer, that can take the text as an input and give out its summary as an output. The advent of the transformers in ML and NLP models has made the task easier and more efficient. After having this much information, we have implemented an NLP model of text summarizer, that makes use of the T5 transformer to summarize the input text.

II. DATA PREPROCESSING AND EDA

Data Source and Encoding: The data for this research consists of news articles and their summaries. The data contained six columns: author, date, headlines, read more, text, and ctext. We processed the data to retain only the text and ctext columns. These were subsequently renamed to summary and text and we removed all the rows that contain missing values in either summary or text column. The data set contains total of 4514 rows and 2 columns.



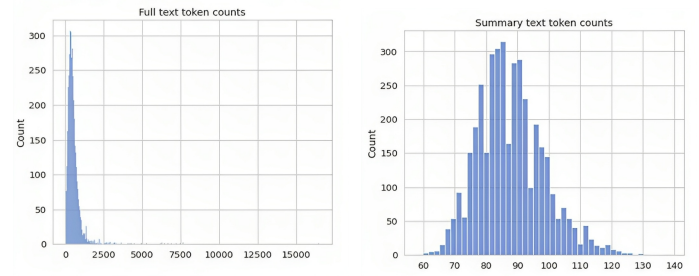
(a) Initial Dataset



(b) Cleaned Dataset

Fig. 1

The encoding used was "latin," a commonly used format for textual data that ensures accurate character representation and avoids potential errors during processing. The T5 Tokenizer used here in the model maps words in the data set to a unique numerical IDs.



(a) Text Token Counts

(b) Summary Text Token Counts

Fig. 2

We are plotting histograms of the token counts for both the full texts and their corresponding summaries in a training dataset for a text summarization task and analyzing in each text and summarizing the distribution of these counts using histograms. Each histogram shows: On the X-axis: The range of possible token counts. On the Y-axis: The relative frequency of texts with each token count.

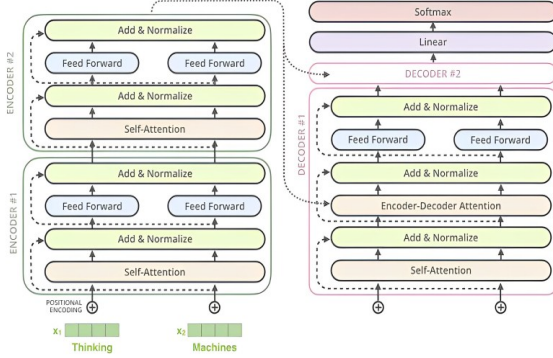
III. TRANSFORMER ARCHITECTURE AND MODEL

Our model is based on the T5 transformer, with the encoder and decoder blocks. T5 differs from other transformer based models like BERT and GPT on the blocks used. BERT and GPT uses either of the encoder or the decoder block, whereas the T5 model makes use of both the blocks. T5's ability to contextualize (which depends on self-attentional layers) and generate coherent summaries make it a better choice.

Encoder: The encoder block is concerned with taking in and comprehending the input text. In fact, it involves accessing embeddings for each input word/sentence (or token).

Decoder: It takes the output embeddings from the encoder at each respective step and generates the final output with each subsequent step. First, the input text is tokenized and the tokens are then fed into the encoders which modifies them and forward it to the decoders. But there is a difference - instead of the CNN, T5 encoders and decoders have these three layers - feedforward, add & normalize and self-attention layers. Self-attention determines how much a word is linked to other words in the sentence. It is obvious that

more self-attention weight matrices will enable the encoder to capture more range of dependencies. Feedforward layers do two linear transformations separated by a non-linear activation function. The purpose of this layer is to add non-linearity to the model and to enable it to capture more complex patterns and links within the text. Add and Normalize layer between each sublayer captures their output and adds them again to the input of the sublayers, leading to better performance of the model. After the addition has been done, layer normalization is done which normalizes the activation across the feature dimension. This ensure that input to each layer is mean-centred with unit variance.



IV. TRAINING AND TESTING

1) Training

First, the data is split into training data(80%) and testing data(20%). Then we pre-processed the text and summary pairs using the T5Tokenizer. We used separate a function to generate encoded representations of both the text and the summary. Further, NewsSummaryDataset class was used to gain access and retrieve the individual datapoints, while preparing them in a format that is readily usable by the T5 model for training as well as evaluation. We also initialised the tokenizer for our T5 model. For the training, we have set:

- N EPOCHS = 10
- BATCH SIZE = 4

The next step involved initialising the model and defining how inputs will be processed and output will be given by it. In the next step, we define how much the model will learn from each training batch and loss is calculated as the output of the training step.

2) Validation, Test and Optimization

The next step is validation. The same thing is repeated as in training, but with new datasets. The output loss is calculated, which gives an idea about the efficiency of the model on new data. The learning rate is set to 0.0001 and for optimization, AdamW optimizer is used in order to update model's internal parameter during the training. Then, we used functions to do checkpointing to save the best performance of the model logging to visualize the training process. Finally, with the help of the loading and freezing of the best model, we finalize the model to use it for the

summarization task. It is to be remembered that we have trained our model for extractive summarization, not abstractive one.

V. RESULTS AND ACCURACY

	Precision	Recall	F measure
Rouge 1	0.37089393132223053	0.5217009323665984	0.42218026619874477
Rouge 2	0.18426551098033594	0.263809873090168	0.21066942342702538
Rouge L	0.2726865275666802	0.3865089325215334	0.31117806241078616

To determine the performance of our model, we use a measure called ROUGE-score, which measures how well a generated text (like a summary) matches a reference or a set of reference texts. Two of main types of ROUGE scores:

- **Rouge N:** Measures how many sequences of n consecutive words (n-grams) in the generated text match those in the reference text. ROUGE N is further classified into many types, two of which are **ROUGE 1** and **ROUGE 2**.
- **Rouge L:** Examines the longest common subsequences of words between the generated and reference summaries.

VI. CONCLUSION

Extractive text summarization has a wide range of practical applications in various industries and fields. Here are some specific cases where it can be particularly useful:

- **News Summarization:** Automatically generate short news summaries for websites, mobile apps, and social media platforms. This allows users to quickly grasp the key information without having to read the entire article.
- **Social Media Analysis:** Analyze social media trends and sentiment by automatically summarizing and categorizing posts.

VII. ACKNOWLEDGEMENT

Special thanks to Dr. Animesh Chaturvedi for giving us this opportunity to work on a popular technology such as Text Summarization under Artificial Intelligence.

VIII. SOURCES

- **Eren Kızılrnak and Alparslan Mesri (2023)** Text Summarization: How to Calculate Rouge Score
- **Mingye Wang, Pan Xie, Yao Du and Hiaohui Hu (2023)** T5- Based Model for Abstractive Summarization: A Semi-Supervised Learning Approach with Consistency Loss Functions
- **un-Ping Ng and Viktoria Abrecht (2015)** Better Evaluation with Word Embeddings for ROUGE
- **Haohan Shi and Phillip Wolf (2021)** What Transformers Might Know About the Physical World: T5 and the Origins of Knowledge
- **Khushnuma Grover, Katinder Kaur, Kartikey Tiwari and Parteek Kumar (2021)** Deep Learning Based Question Generation Using T5 Transformer