

TEXT SUMMARIZATION USING DEEP LEARNING
GROUP 3
INCREMENT 1

Team Members :

Hari Vamshi Krishna (11614861)
Manish Reddy Radha Reddy (11518946)
Ramya Sree Konduru (11601633)
Ranjith Kumar Sadafule (11602281)

GitHub Repository of the Project : <https://github.com/thor56/NLPTextSummarization.git>

Goals and Objectives

Motivation:

The main motivation of this project came from preparing summarized notes for exam preparation. Generally, we have our lecture notes, but they are lengthy and has lots of content in it and we really can't use the lecture notes for the exam preparation because, it is covered of small and general examples which help us to understand the concept, repeated concepts with different approach. Let us suppose we have 500 pages of lecture notes, and we can't go through all the 500pages all the time when we are preparing for our exam. So, to overcome this we make a short note of all the important points and convert the 500 pages notes to 10 pages or so depending on the requirement.

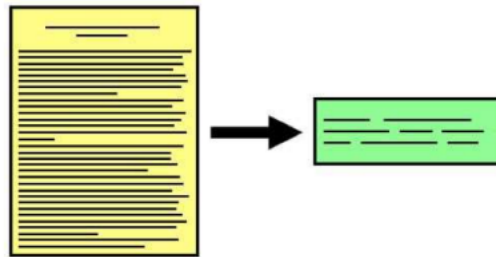


Figure-1: Displaying the Text summarization representation

In the above figure 1, displays the pictorial representation of text summarization, where it shows that 50 lines in one document are summarized into 5 lines.

So, our main aim is to create a text summarization model using Neural network which will provide meaningful sentences without missing the information.

Significance:

The text summarization cannot only be used in the lecture notes summarization, when we look at the broader picture, it can be used for Media monitoring, Newsletters, Search Marketing and SEO, Internal document workflow, financial research, Legal Contract analysis, social media marketing and many more. People always need their work to be reduced. So, in the context of reading and making short notes text summarization will be very useful.

Important aspects of the text are identified by the Deep Learning model and retained and reframed to represent the same content.

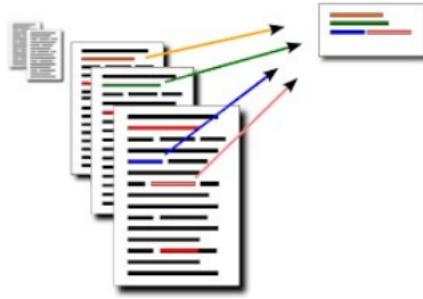


Figure-2: Illustrating the application of Text Summarization.

Objectives:

The main objective of our project is to deliver a text summarization model which performs good when compared to the State of Art models and provide meaningful sentences, without missing any important information. Here, we will be working on different models that are present for text summarization and compare the performance of the model. Using tools and implementations of NLP, the model could obtain consistent output that represents the original context as is but with significantly smaller text content.

Features:

Deep Learning modelling helps in enhancing the identification of keywords in the text, this improvement leads to summarizing the text without missing any important information.

Suppose we have different keywords with the same meaning, then this situation can also be overcome by the deep learning model and provide us best results. The working model will reduce the length of text of given context to smaller text length while maintaining coherent and fluent summary which contain only key points of the text.

INTRODUCTION

Here, we will be performing some of the State of the Art Models for Text summarization and do analysis on these models

These models we perform today include:

1. Transformer.
2. GPT-2

3. BART
4. PEGASUS
5. T-5

Transformer:

A transformer uses a simple encoder-decoder architecture.

GPT-2:

(Generative pre-trained Transformer-2)GPT-2 is a pre-trained transformer model which is pre-trained on large english data, this is pre-trained on raw data only.

BART:

It is an autoencoder for pre-training a sequence to sequence model.

PEGASUS:

It is a State of the Art model for abstractive text summarization which uses an encoder-decoder model for sequence to sequence learning.

T5 Summarization:

Text to Text transfer transformer which is based on the transformer model, it uses the text-to-text approach.

Then we will be evaluating the performance of all the models using two metrics ROUGE and BLEU, one to measure precision and one to measure recall, then we perform our model on the test set of the dataset and compute the score.

BACKGROUND WORK:

Text summarization is a process of reducing the words of a given input document without the actual loss of any information from the original document(Creating a summary of the document).

Text summarization is an essential task in NLP, it is of two types:

1. Abstractive based Text Summarization
2. Extractive based text summarization.

Abstractive Based Text Summarization:

In this Abstractive method it first builds an internal grammar structure and then uses natural language processing techniques to create the summary, the created summary thus may contain some words which we might not have seen in the original document but the end meaning is the same.

Extractive Based Text Summarization:

In contrast to the Abstractive based summary technique, extractive based summarization selects a chunk of words or sentences from the original document and then forms a summary from that. The important task in extractive based summarization is identifying key sentences in the document and making use of them to create a summary.

Related work

The current situation has been concluded from the existing work that has been done on Amazon product reviews to summarize. Several approaches have been devised for the summarization in general, summarizing lecture notes should have similar treatment on the dataset and implementation to summarize the notes. From summarizing customer reviews on online platforms to summarizing Tweets and movie storylines have been shown have great accuracy, this has been the right motivation to develop a model to be able to summarize lecture notes to save time for students while going through the lecture notes.

Dataset

The dataset has been collected from Kaggle and here the data has been segregated into paragraphs from the lecture. There are various attributes in this dataset such as Title, Department, Field of study, Description and conclusion. All these attributes are properly analyzed for summarizing on point with highest accuracy.

Detailed design of Features

The core design of the model is to summarize and shorten the lecture notes while retaining the amount of information initially conveyed.

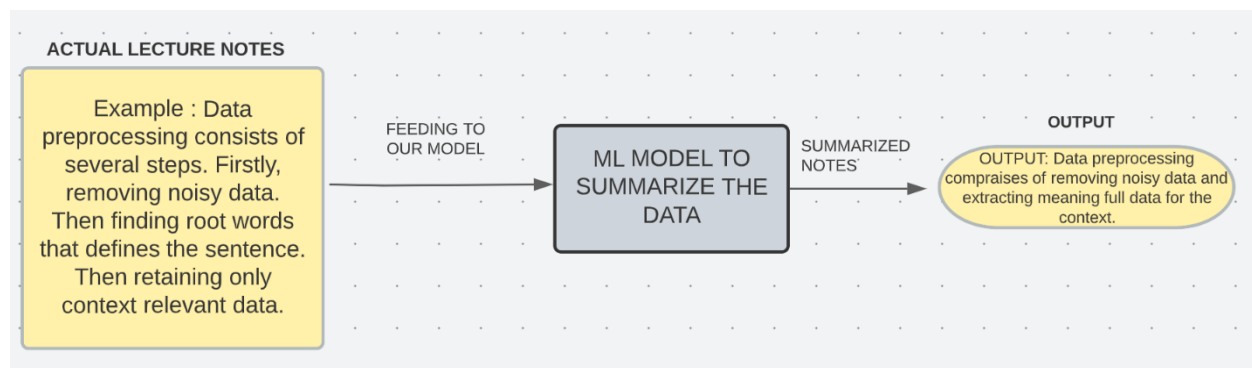


Figure 3: Core feature of the Model

(Source: Self-developed)

The model uses Machine Learning implementation to process and analyse the data in the lecture notes and based on training, It produces shortened notes. It is integrated with several modeling steps to reproduce the exact same meaning of the context.

Analysis

It was initially analyzed that, based on different subjects, the accuracy of the model differs slightly. However, that is a problem for the final increment as the current state of the model still attains great accuracy and outcome. Not a lot of data is required to train the model but significant amount is required to be able keep the summarizing in relevancy to the context.

Implementation

Initially we perform the data preprocessing steps in order to be able to work with the only important data for building the model upon.

Drop Duplicates and NA values

```
data.drop_duplicates(subset=['Text'],inplace=True)#dropping duplicates
data.dropna(axis=0,inplace=True)#dropping na
```

Information about dataset

Let us look at datatypes and shape of the dataset

```
data.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 162834 entries, 0 to 199999
Data columns (total 10 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Id                     162834 non-null int64
1   ProductId             162834 non-null object
2   UserId                162834 non-null object
3   ProfileName           162834 non-null object
4   HelpfulnessNumerator   162834 non-null int64
5   HelpfulnessDenominator 162834 non-null int64
6   Score                 162834 non-null int64
7   Time                  162834 non-null int64
8   Summary               162834 non-null object
9   Text                  162834 non-null object
dtypes: int64(5), object(5)
memory usage: 13.7+ MB
```

Figure 4: About the dataset and preprocessing

(Source: Self-developed)

Our model follows Sequence to sequence modeling to process any sequential information.

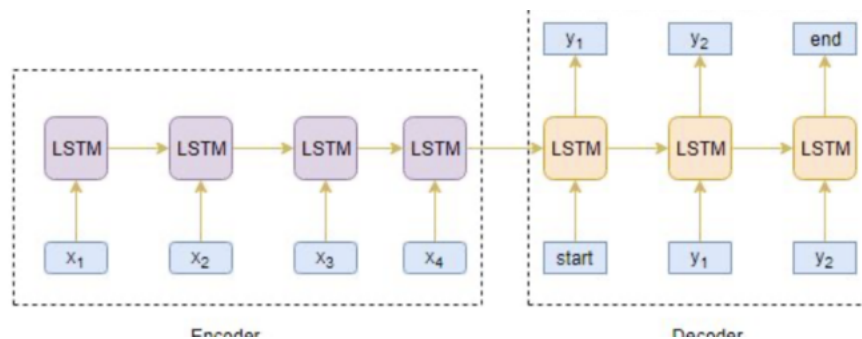


Figure 5: Principle working of the model

(Source: Self-developed)

The Seq 2 Seq model constitute of 2 major components which are Encoder and Decoder phases. Either Gated Recurrent Neural Network or Long Short Term Memory (LSTM) is preferred for encoding and decoding components. While we choose LSTM

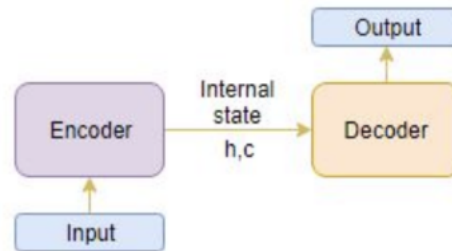


Figure 6: Illustration of the underlying working of Model
(Source: Self-developed)

The LSTM Encoder reads the entire input and passes one word as a token each time from which the context is extracted. Then, LSTM Decoder predicts same sequences with certain offset and it is also taught to anticipate following words.

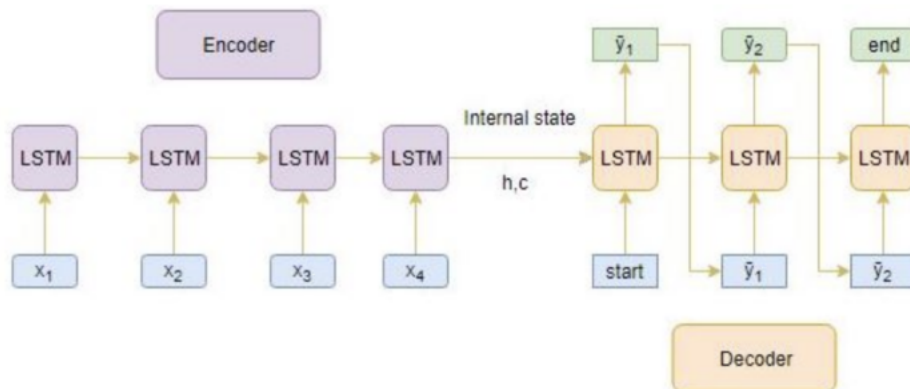


Figure 7: Formation of comprehension
(Source: Self-developed)

The initial breakdown and processing of our model on the dataset is visualized as follows:

Model: "model"			
Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 30)]	0	[]
embedding (Embedding)	(None, 30, 100)	1146600	['input_1[0][0]']
lstm (LSTM)	[(None, 30, 300), (None, 300), (None, 300)]	481200	['embedding[0][0]']
input_2 (InputLayer)	[(None, None)]	0	[]
lstm_1 (LSTM)	[(None, 30, 300), (None, 300), (None, 300)]	721200	['lstm[0][0]']
embedding_1 (Embedding)	(None, None, 100)	296500	['input_2[0][0]']
lstm_2 (LSTM)	[(None, 30, 300), (None, 300), (None, 300)]	721200	['lstm_1[0][0]']
lstm_3 (LSTM)	[(None, None, 300), (None, 300), (None, 300)]	481200	['embedding_1[0][0]', 'lstm_2[0][1]', 'lstm_2[0][2]']
attention_layer (AttentionLayer)	((None, None, 300), (None, None, 30))	180300	['lstm_2[0][0]', 'lstm_3[0][0]']
concat_layer (Concatenate)	(None, None, 600)	0	['lstm_3[0][0]', 'attention_layer[0][0]']
time_distributed (TimeDistributed)	(None, None, 2965)	1781965	['concat_layer[0][0]']
=====			
Total params: 5,810,165			
Trainable params: 5,810,165			
Non-trainable params: 0			

Figure 8: LSTM and Attention layer breakdown

(Source: Self-developed)

The implementation involves attention layer which is used to define the scope of the word or phrase with respect to rest of the context. We have used about 10 Epoch to train our model.

```
history=model.fit([x_tr,y_tr[:, :-1]], y_tr.reshape(y_tr.shape[0],y_tr.shape[1], 1)[:,:1:], epochs=10,callbacks=[es],batch_size=128)

Epoch 1/10
604/604 [=====] - 657s 1s/step - loss: 2.8004 - val_loss: 2.5370
Epoch 2/10
604/604 [=====] - 655s 1s/step - loss: 2.4766 - val_loss: 2.3749
Epoch 3/10
604/604 [=====] - 636s 1s/step - loss: 2.3480 - val_loss: 2.2856
Epoch 4/10
604/604 [=====] - 636s 1s/step - loss: 2.2699 - val_loss: 2.2358
Epoch 5/10
604/604 [=====] - 636s 1s/step - loss: 2.2142 - val_loss: 2.1954
Epoch 6/10
604/604 [=====] - 644s 1s/step - loss: 2.1674 - val_loss: 2.1604
Epoch 7/10
604/604 [=====] - 648s 1s/step - loss: 2.1296 - val_loss: 2.1364
Epoch 8/10
604/604 [=====] - 656s 1s/step - loss: 2.0967 - val_loss: 2.1183
Epoch 9/10
604/604 [=====] - 660s 1s/step - loss: 2.0679 - val_loss: 2.0968
Epoch 10/10
604/604 [=====] - 642s 1s/step - loss: 2.0423 - val_loss: 2.0852
```

Figure 9: Model training

(Source: Self-developed)

Preliminary results

The initial results seem promising with close to actual summary of the text. We have generated many summarized out of which some are produced below

```
In [55]: for i in range(0,100):
          print("Review:",seq2text(x_tr[i]))
          print("Original summary:",seq2summary(y_tr[i]))
          print("Predicted summary:",decode_sequence(x_tr[i].reshape(1,max_text_len)))
          print("\n")
```

```
Review: policy use non gmo products possible cannot guarantee product produce
d non gmo corn low prices majority corn produced us gmo assume gmo product gr
eat amazon return policy food items stuck
Original summary: warning they use corn for this product
1/1 [=====] - 0s 353ms/step
1/1 [=====] - 0s 52ms/step
1/1 [=====] - 0s 39ms/step
1/1 [=====] - 0s 45ms/step
1/1 [=====] - 0s 40ms/step
1/1 [=====] - 0s 39ms/step
Predicted summary: not worth the money
```

```
In [55]: for i in range(0,100):
          print("Review:",seq2text(x_tr[i]))
          print("Original summary:",seq2summary(y_tr[i]))
          print("Predicted summary:",decode_sequence(x_tr[i].reshape(1,max_text_len)))
          print("\n")
```

```
Predicted summary: not the best
```

```
Review: special treat babies pricey would think dogs would like feel look lik
e vegetable shaped crayons full sorts goodies although feeding grain free die
t first ingredient hydrolyzed wheat protein
Original summary: my dogs love love love
1/1 [=====] - 0s 369ms/step
1/1 [=====] - 0s 38ms/step
1/1 [=====] - 0s 38ms/step
1/1 [=====] - 0s 37ms/step
1/1 [=====] - 0s 39ms/step
1/1 [=====] - 0s 37ms/step
Predicted summary: my dogs love these
```

```
Review: gluten free oatmeal delicious always think like apple cinnamon flavor
better fast shipping exceptional pricing
Original summary: delicious
1/1 [=====] - 0s 350ms/step
1/1 [=====] - 0s 39ms/step
1/1 [=====] - 0s 37ms/step
1/1 [=====] - 0s 43ms/step
1/1 [=====] - 0s 39ms/step
1/1 [=====] - 0s 39ms/step
Predicted summary: great gluten free snack
```

As we can see, the comprehended text already seem pretty close to the actual expected outcome.

Project management

Work completed:

Description

Dataset has been collected from Kaggle and preprocessing has been performed on it. Basing on the LSTM modeling technique, a basic ML model has been developed which processes by breaking down individual text and predicting next in the sequence and matching it with the actual following word. This is called Sequence to Sequence modeling. The data is feed to Encoder which breaks down the phrase into words and send them as input to the decoder which predicts the next word based on the key context. This also involves custom attention layer which is responsible to maintain relevancy factor with respect to the key context (distinguishing whether the phrase is local or global context).

Responsibility :

Manish Reddy Radha Reddy : Analysed the existing models for similar goals and decided the suitable approach. Furthermore, is working on increasing the accuracy of the model based on the State-of-the-Art model available out there.

Ramya Sree Konduru : Gathered the Dataset and preprocessed the data for the training to be carried out. Documented the teamwork and process.

Hari Vamshi Krishna : Defined the suitable Model and approach to build the model. Tried and built the model based on ML approach

Ranjith Kumar Sadafule : Performed model training and model testing and validation on the processed dataset and produced the preliminary results. Documented the teamwork and process.

Contribution:

Manish Reddy Radha Reddy : 25%

Ramya Sree Konduru : 20%

Hari Vamsi Krishna Samayamantri :30%

Ranjith Kumar Sadafule :25%

Work To Be Completed :**Description :**

The initial implementation has worked pretty well in favour our model compared to the actual output. Although there can be a lot of improvement when compared to the existing State-of-the-Art model that perform similar operations. We are working towards increasing the accuracy of out model to be on par with the SOTA models as mention.

Responsibility and Contribution:

Manish Reddy Radha Reddy : Adding improvements to the existing model to attain greater accuracy.

Ramya Sree Konduru : Make potential changes to data preprocessing steps to eliminate some more irrelevant data and retain context-familiar data

Hari Vamshi Krishna : Analyze differences in key working mechanism of the State of the art model and our models to identify key performance pull backs.

Ranjith Kumar Sadafule : Run multiple combinations of model training and testing

References

- Anand, C., 2021. Comparison of stock price prediction models using pre-trained neural networks. *Journal of Ubiquitous Computing and Communication Technologies (UCCT)*, 3(02), pp.122-134.
- Hu, Z., Zhao, Y. and Khushi, M., 2021. A survey of forex and stock price prediction using deep learning. *Applied System Innovation*, 4(1), p.9.
- Mehtab, S. and Sen, J., 2020. Stock price prediction using convolutional neural networks on a multivariate timeseries. *arXiv preprint arXiv:2001.09769*.
- Obthong, M., Tantisantiwong, N., Jeamwatthanachai, W. and Wills, G., 2020. A survey on machine learning for stock price prediction: algorithms and techniques.

TEXT SUMMARIZATION USING DEEP LEARNING
GROUP 3
INCREMENT 2

Team Members :

Hari Vamsi Krishna (11614861)
Manish Reddy Radha Reddy (11518946)
Ramya Sree Konduru (11601633)
Ranjith Kumar Sadafule (11602281)

GitHub Repository of the Project : <https://github.com/thor56/NLPTextSummarization.git>

INTRODUCTION

Here, we will be performing some of the State of the Art Models for Text summarization and do analysis on these models

These models we perform today include:

6. Transformer.
7. GPT-2
8. BART
9. PEGASUS
10. T-5

Transformer:

A transformer uses a simple encoder-decoder architecture.

GPT-2:

(Generative pre-trained Transformer-2)GPT-2 is a pre-trained transformer model which is pre-trained on large english data, this is pre-trained on raw data only.

BART:

It is an autoencoder for pre-training a sequence to sequence model.

PEGASUS:

It is a State of the Art model for abstractive text summarization which uses an encoder-decoder model for sequence to sequence learning.

T5 Summarization:

Text to Text transfer transformer which is based on the transformer model, it uses the text-to-text approach.

Then we will be evaluating the performance of all the models using two metrics ROUGE and BLEU, one to measure precision and one to measure recall, then we perform our model on the test set of the dataset and compute the score.

BACKGROUND WORK:

Text summarization is a process of reducing the words of a given input document without the actual loss of any information from the original document(Creating a summary of the document).

Text summarization is an essential task in NLP, it is of two types:

3. Abstractive based Text Summarization
4. Extractive based text summarization.

Abstractive Based Text Summarization:

In this Abstractive method it first builds an internal grammar structure and then uses natural language processing techniques to create the summary, the created summary thus may contain some words which we might not have seen in the original document but the end meaning is the same.

Extractive Based Text Summarization:

In contrast to the Abstractive based summary technique, extractive based summarization selects a chunk of words or sentences from the original document and then forms a summary from that. The important task in extractive based summarization is identifying key sentences in the document and making use of them to create a summary.

MODEL

TRANSFORMER:

Architecture:

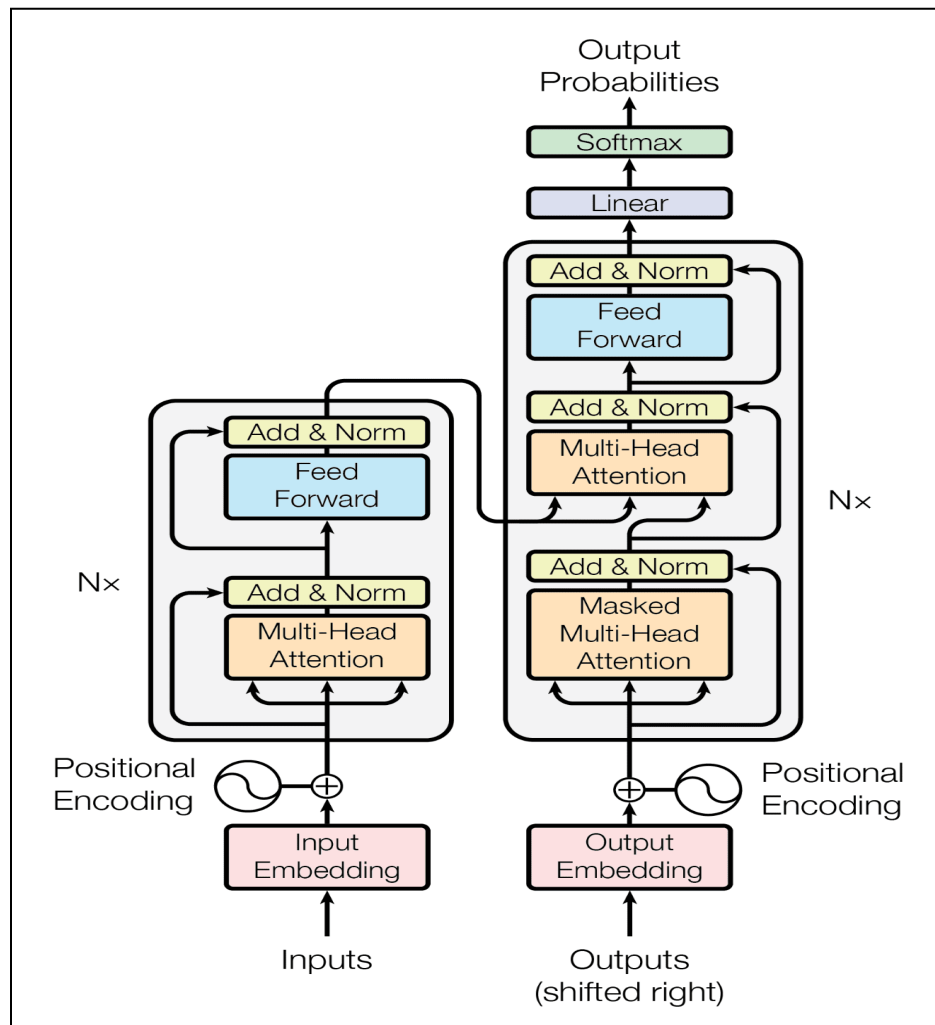


Fig: Sample lecture notes from the dataset

(Source: Internet)

- Transformer follows an encoder-decoder architecture and generates an output

- Task of the encoder is to map the input sequence to another contiguous sequence which is then passed to the decoder.
- The decoder which we can see on the right side receives the output from the encoder along with the output from the decoder at the previous timestep and generates an output sequence.

ENCODER:

- Encoder consists of a stack of identical layers and each layer has 2 sub layers
- The first layer implements self- attention mechanism
- The second layer is connected to a feed-forward neural network which has a RELU(Rectified Linear Unit) activation function.
- All the layers in the transformer apply the same linear transformations to all the phrases in the input sequence but each layer applies different weights and biases.

DECODER:

- The decoder is similar to the encoder, that is it has a stack of identical layers but each layer has 3 sub layers.
- The first sublayer gets the output from the decoder and combines it with the information about the position and implements attention over it.
- The second sub layer applies a multi-head self attention mechanism, this multi-head mechanism gets keys and values from the output of the encoder.
- The third layer is connected to a feed-forward neural network which has a RELU(Rectified Linear Unit) activation function similar to the second layer in the encoder.

All the models which we performed in the code are based on this transformer model.

WORKFLOW DIAGRAM:

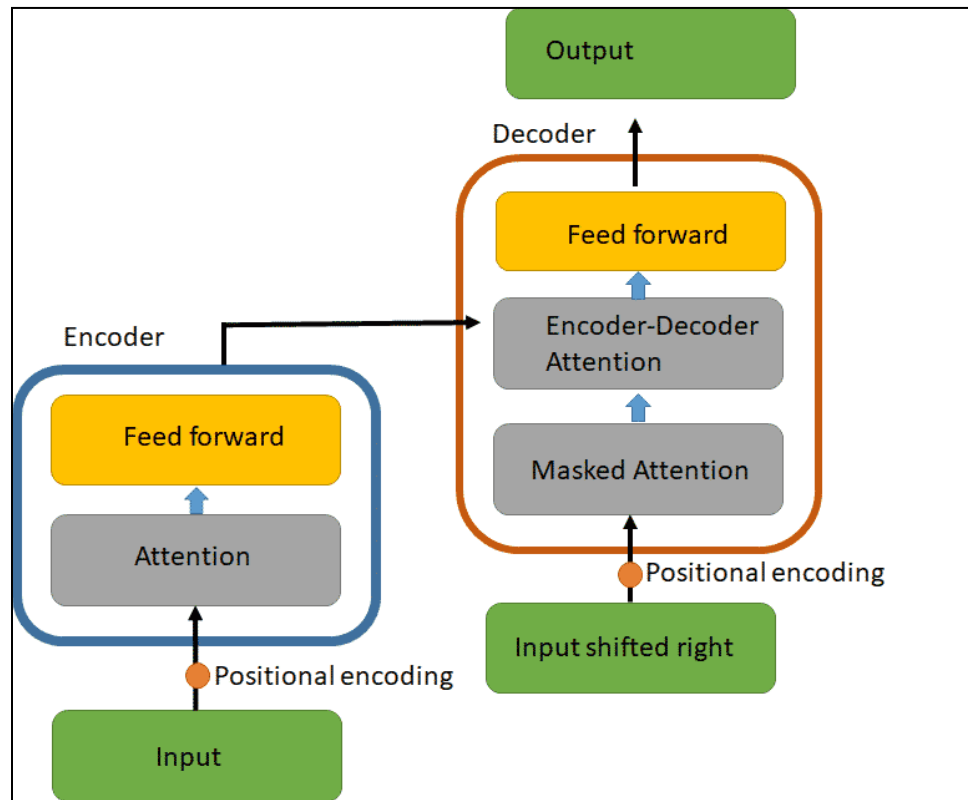


Fig : Workflow diagram

(Source: Internet)

Basic workflow diagram of the transformer architecture which was shown above, since the architecture is complex we tried to make the workflow diagram simple in order to explain it easily.

The input document goes to the encoder and that's where positional encoding happens as we mentioned it has 2 sublayers one with self-attention and other connected to a feed-forward neural network.

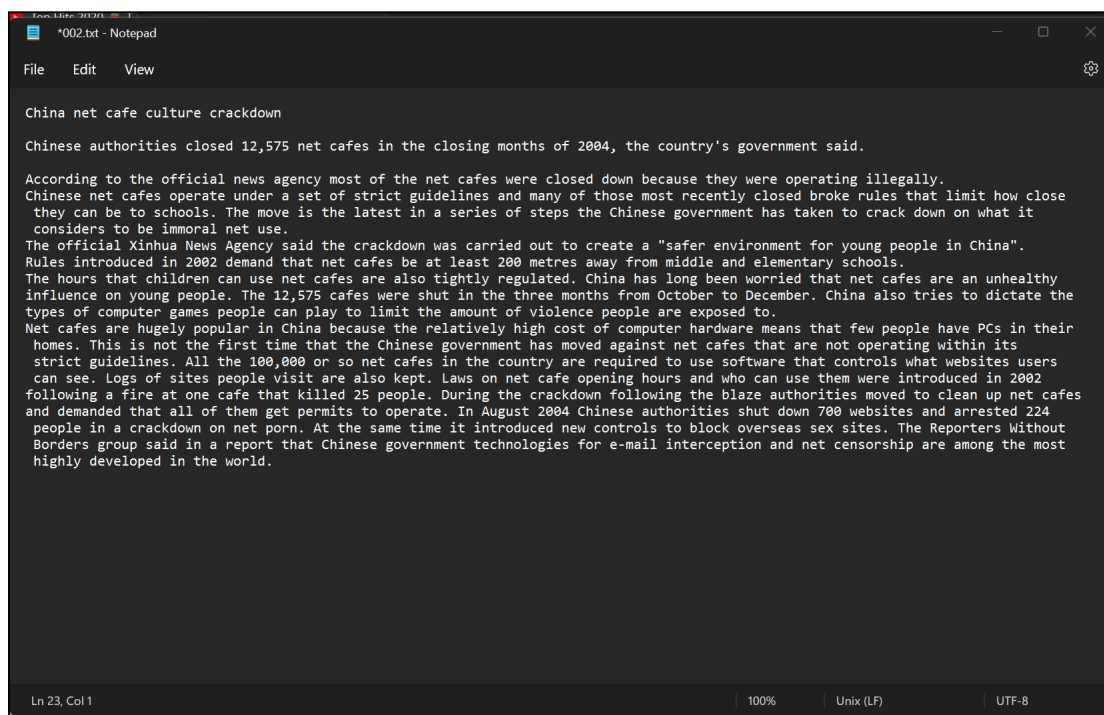
Positional encoding is the method where it adds a sequence to the input sequence.

The output from the encoder is shifted to the decoder attention which then combines with the masked attention and is fed to the 3rd sublayer of the decoder a feed forward neural network and we get the output.

Dataset

Detailed description of Dataset

- The dataset is obtained from Kaggle platform, It consists of paragraphs from lectures and consists of attributes Title, Paragraphs from the lecture, Department, Field of study, Description and Conclusion. These data are further pre-processing and cleaned so It could be used for performing analysis.
- The dataset contains actual lecture notes from lectures shared by professors for their classes, this way we can train the model to make it work on any lecture notes. Here are a few samples of lecture notes from the dataset.



```
*002.txt - Notepad
File Edit View

China net cafe culture crackdown

Chinese authorities closed 12,575 net cafes in the closing months of 2004, the country's government said.

According to the official news agency most of the net cafes were closed down because they were operating illegally. Chinese net cafes operate under a set of strict guidelines and many of those most recently closed broke rules that limit how close they can be to schools. The move is the latest in a series of steps the Chinese government has taken to crack down on what it considers to be immoral net use.

The official Xinhua News Agency said the crackdown was carried out to create a "safer environment for young people in China". Rules introduced in 2002 demand that net cafes be at least 200 metres away from middle and elementary schools. The hours that children can use net cafes are also tightly regulated. China has long been worried that net cafes are an unhealthy influence on young people. The 12,575 cafes were shut in the three months from October to December. China also tries to dictate the types of computer games people can play to limit the amount of violence people are exposed to.

Net cafes are hugely popular in China because the relatively high cost of computer hardware means that few people have PCs in their homes. This is not the first time that the Chinese government has moved against net cafes that are not operating within its strict guidelines. All the 100,000 or so net cafes in the country are required to use software that controls what websites users can see. Logs of sites people visit are also kept. Laws on net cafe opening hours and who can use them were introduced in 2002 following a fire at one cafe that killed 25 people. During the crackdown following the blaze authorities moved to clean up net cafes and demanded that all of them get permits to operate. In August 2004 Chinese authorities shut down 700 websites and arrested 224 people in a crackdown on net porn. At the same time it introduced new controls to block overseas sex sites. The Reporters Without Borders group said in a report that Chinese government technologies for e-mail interception and net censorship are among the most highly developed in the world.
```

Fig : Sample lecture notes from the dataset

(Source: Self-developed)

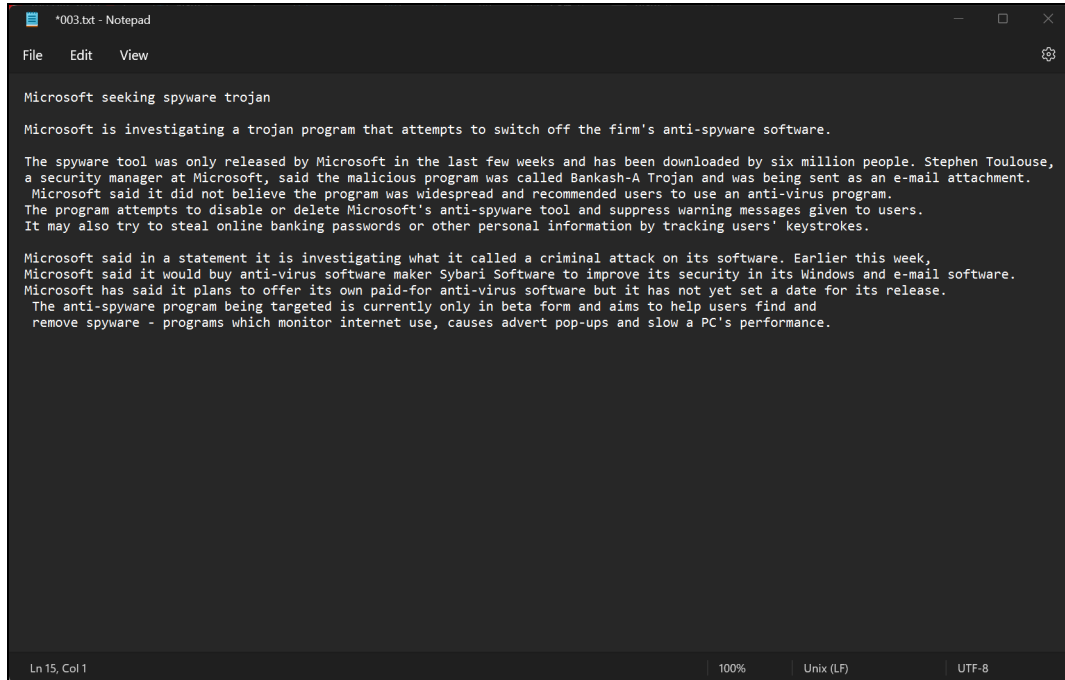


Fig : Sample lecture notes from the dataset

(Source: Self-developed)

Detail design of Features with diagram

- The dataset consists of multiple attributes which are Title, Department, Field of Study, Description and Conclusion. The Title is the topic of the lecture being discussed.
- The Department field is the domain of the subject like Computer Science or Physics. The Field of Study is the discrete topic under the department like Algorithms for Computer Science or Thermodynamics for Physics.
- Description is the actual content of the lecture notes used to describe the entire discussion. Conclusion is the summary of the entire lecture with key points being highlighted. The structure of the dataset is illustrated below

STRUCTURE OF THE DATASET				
TITLE	DEPARTMENT	FIELD OF STUDY	DESCRIPTION	CONCLUSION
THE TOPIC WHICH IS COVERED IN THIS LECTURE NOTES	WHICH DEPARTMENT OF THE SCHOOL THE CLASS BELONGS TO	THE MAINSTREAM TOPIC OR SUBJECT OF THE LECTURE	THE ACTUAL LECTRUER NOTES	THE END POINT OF THE LECTURE THAT THE LECTURE EXPLAINS ABOUT

Fig : Structure of the dataset

(Source: Self-developed)

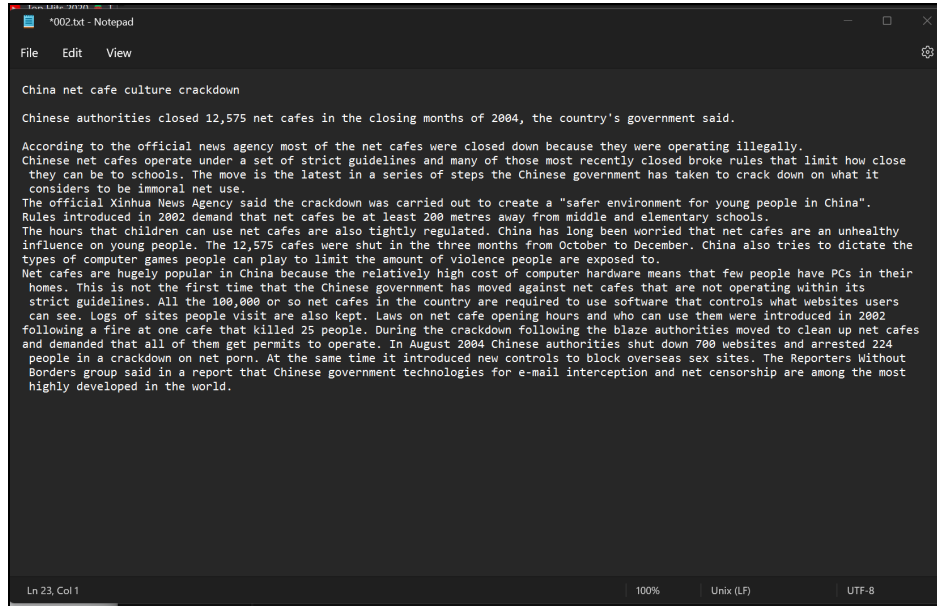
The Title attribute can be used to niche down the context topic of the lecture from a broad perspective. An idea can be achieved over the topic of the lecture. The description is the primary attribute that plays a major role in summarizing the lecture. This contains the actual content from the lecture and needs to be processed by the model for summarization.

Analysis of data

Data Pre-processing

The dataset obtained will not be ideal to train our model so we have to perform some pre-processing steps and transform the data to better suit our model and analysis. For instance, the data contains several paragraphs but we have to identify and separate the attributes of the data into Title, Description, Department and Conclusion.

For this we used python's text processing capabilities to identify the defined order of these attributes in the data. We pre-processed the data and stored it in a separate folder. The raw state of data is illustrated below.

A screenshot of a Notepad window titled "002.txt - Notepad". The text inside the window is as follows:

China net cafe culture crackdown

Chinese authorities closed 12,575 net cafes in the closing months of 2004, the country's government said.

According to the official news agency most of the net cafes were closed down because they were operating illegally. Chinese net cafes operate under a set of strict guidelines and many of those most recently closed broke rules that limit how close they can be to schools. The move is the latest in a series of steps the Chinese government has taken to crack down on what it considers to be immoral net use.

The official Xinhua News Agency said the crackdown was carried out to create a "safer environment for young people in China". Rules introduced in 2002 demand that net cafes be at least 200 metres away from middle and elementary schools. The hours that children can use net cafes are also tightly regulated. China has long been worried that net cafes are an unhealthy influence on young people. The 12,575 cafes were shut in the three months from October to December. China also tries to dictate the types of computer games people can play to limit the amount of violence people are exposed to.

Net cafes are hugely popular in China because the relatively high cost of computer hardware means that few people have PCs in their homes. This is not the first time that the Chinese government has moved against net cafes that are not operating within its strict guidelines. All the 100,000 or so net cafes in the country are required to use software that controls what websites users can see. Logs of sites people visit are also kept. Laws on net cafe opening hours and who can use them were introduced in 2002 following a fire at one cafe that killed 25 people. During the crackdown following the blaze authorities moved to clean up net cafes and demanded that all of them get permits to operate. In August 2004 Chinese authorities shut down 700 websites and arrested 224 people in a crackdown on net porn. At the same time it introduced new controls to block overseas sex sites. The Reporters Without Borders group said in a report that Chinese government technologies for e-mail interception and net censorship are among the most highly developed in the world.

Ln 23, Col 1 100% Unix (LF) UTF-8

Fig : Data before pre-processing
(Source: Self-developed)

Several other factors have to be considered so as to handle Numerical data, Formulae and other equations.

Implementation

- Now let's go step by step on how we have implemented the state of the art models.

Step 1:

- install the transformers and set-up everything for our code.
- We pip install the transformers model since it is not available in google colab and import the necessary libraries like numpy,pandas auto encoder, auto tokenizer and NLTK
- Then we create a pipeline where we will be implementing all our pre-trained models.


```
!pip install transformers[sentencepiece] datasets sacrebleu rouge_score py7zr -q
```

5.5 MB	31.7 MB/s
451 kB	61.2 MB/s
118 kB	81.0 MB/s
65 kB	4.8 MB/s
182 kB	65.0 MB/s
212 kB	3.6 MB/s
132 kB	54.1 MB/s
127 kB	74.1 MB/s
50 kB	6.6 MB/s
378 kB	66.3 MB/s
139 kB	83.3 MB/s
94 kB	3.5 MB/s
357 kB	77.3 MB/s
2.3 MB	64.5 MB/s
7.6 MB	66.7 MB/s
1.3 MB	61.3 MB/s

Building wheel for rouge-score (setup.py) ... done

```
from transformers import pipeline, set_seed

import matplotlib.pyplot as plt

import pandas as pd
from datasets import load_dataset, load_metric
from transformers import AutoModelForSeq2SeqLM, AutoTokenizer

import pandas as pd
import numpy as np

import nltk
from nltk.tokenize import sent_tokenize

nltk.download("punkt")

[nltk data] Downloading package punkt to /root/nltk data...
```

Fig : Using Transformer model

(Source: Self-developed)

STEP 2:

- We will be printing the sample data and determine the length of the text and we will print the summary of the text and calculate its length so that we can get an idea of how long the predicted summary of our model should be.

```

▶ #In this step we will be printing the sample text data and the length of the t
sample = dataset["train"][1]
print(f"""
Article (excerpt of 500 characters, total length: {len(sample["article"])}):
""")
print(sample["article"][:500])
print(f'\nSummary (length: {len(sample["highlights"])}):')
print(sample["highlights"])

Article (excerpt of 500 characters, total length: 4051):

Editor's note: In our Behind the Scenes series, CNN correspondents share their

Summary (length: 281):
Mentally ill inmates in Miami are housed on the "forgotten floor"
Judge Steven Leifman says most are there as a result of "avoidable felonies"
While CNN tours facility, patient shouts: "I am the son of the president"
Leifman says the system is unjust and he's fighting for change .

```

Fig : Sample data
(Source: Self-developed)

STEP 3:

- Now we will collect the generated summaries of each model in a dictionary, we will create a function to combine the multiple lines in text data and return its summary.

```

[ ] sample_text = dataset["train"][1]["article"][:1000]

# We'll collect the generated summaries of each model in a dictionary
summaries = {}

[ ] # creating a function to combine the three line in a text data and returns the combined text
def baseline_summary_three_sent(text):
    return "\n".join(sent_tokenize(text)[:3])

[ ] summaries['baseline'] = baseline_summary_three_sent(sample_text)

summaries['baseline']

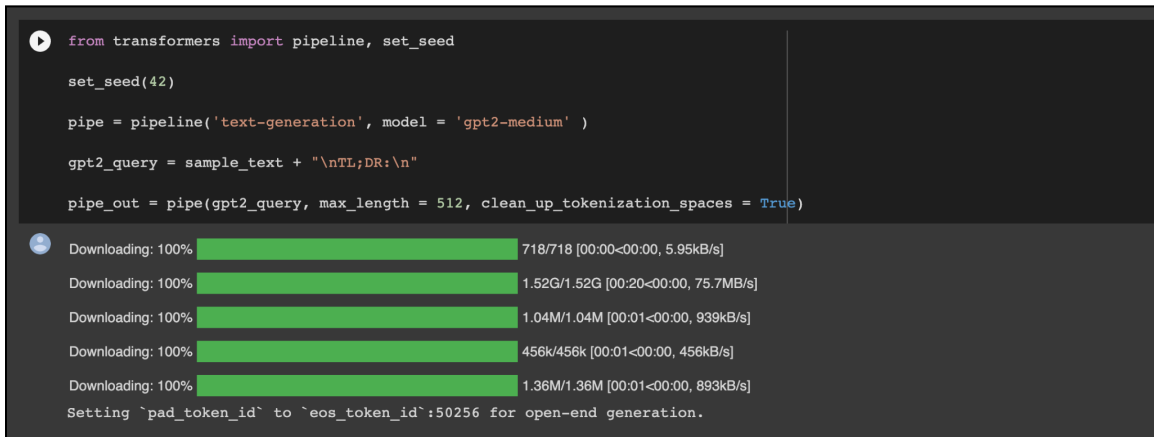
'Editor\'s note: In our Behind the Scenes series, CNN correspondents share their experiences in covering news and analyze the stories behind the events.\nHere, Soledad O\'Brien takes users inside a jail where many of the inmates are mentally ill. An inmate housed on the "forgotten floor," where many mentally ill inmates are housed in Miami before trial.\nMIAMI, Florida (CNN) -- The ninth floor of the Miami-Dade pretrial detention facility is dubbed the "forgotten floor."'

```

Fig :Generated sumamry
(Source: Self-developed)

STEP 4:

- Now we use GPT2 to perform text summarization on our dataset we do that by first creating a text generation pipeline and then load the pre-trained GPT2 model into the pipeline because training GPT2 is exhausting and requires a lot of computational power we use a pre-trained model.
- By appending the TL,DR at the end of the input text to the GPT2 it will help it in generating text. here TL="Too Long" and DR="Didn't read"



```
from transformers import pipeline, set_seed

set_seed(42)

pipe = pipeline('text-generation', model = 'gpt2-medium' )

gpt2_query = sample_text + "\nTL;DR:\n"

pipe_out = pipe(gpt2_query, max_length = 512, clean_up_tokenization_spaces = True)
```

Downloading: 100% 718/718 [00:00<00:00, 5.95kB/s]

Downloading: 100% 1.52G/1.52G [00:20<00:00, 75.7MB/s]

Downloading: 100% 1.04M/1.04M [00:01<00:00, 939kB/s]

Downloading: 100% 456k/456k [00:01<00:00, 456kB/s]

Downloading: 100% 1.36M/1.36M [00:01<00:00, 893kB/s]

Setting 'pad_token_id' to 'eos_token_id':50256 for open-end generation.

Fig : Applying GTP-2
(Source: Self-developed)

STEP 5:

- Now we print out the text in the dataset before running it in our model, and then print the summarized text and then we append it to the empty dictionary.
- GPT2 is an unsupervised deep learning transformer based language model it helps in getting better summarized text with minimal loss.

```

#printing the given text in the dataset
pipe_out

[{'generated_text': 'Editor\\'s note: In our Behind the Scenes series, CNN correspondents share their experiences in covering news and analyze the stories behind the events. Here, Soledad O\\Brien takes users inside a jail where many of the inmates are mentally ill. An inmate housed on the "forgotten floor," where many mentally ill inmates are housed in Miami before trial. MIAMI, Florida (CNN) -- The ninth floor of the Miami-Dade pretrial detention facility is dubbed the "forgotten floor." Here, inmates with the most severe mental illnesses are incarcerated until they\\'re ready to appear in court. Most often, they face drug charges or charges of assaulting an officer --charges that Judge Steven Leifman says are usually "avoidable felonies." He says the arrests often result from confrontations with police. Mentally ill people often won\\'t do what they\\'re told when police arrive on the scene -- confrontation seems to exacerbate their illness and they become more paranoid, delusional, and less likely to follow dir\\n\\n\\nMIAMI-DADE, Florida | April 13, 2012 -- Some inmates are locked up in solitary confinement, where they must be isolated from the world for six months before they\\'re released onto the general jail population to participate in other inmates. Others are housed in "theforgotten floor," the top floor of the pretrial facility that is also used by criminal offenders. What makes these inmates separate is their medical needs. They\\'re given special medications to treat their mental illness, but it\\'s only in those cases where necessary to take certain medications themselves for the very dangerous condition. These medications can lead to anaphylaxis or dangerous reactions if taken by people who are on them alone. A mentally ill person like John Brown, a convicted felon with severe mental illness. He spends four months in the "forgotten floor" and is charged with being a felon, disorderly conduct, assault with a deadly weapon (a gun), possession of child pornography, resisting arrest, and possession of marijuana. Brown was in the second floor when he fell asleep in bed. On April 11, his bed fell over and he was arrested and held until jail officials could check on him. After his detention, Brown refused to return to court the next morning to answer the charges. "I won\\'t be back at all," Brown said. "There will be no new hearings or there will be no court." This episode will air on January 18, 2013 on CNN\\'s Inside Story.'}]

[ ] #printing the summarized text in the dataset.
pipe_out[0][\"generated_text\"][:len(gpt2_query) :]

'MIAMI-DADE, Florida | April 13, 2012 -- Some inmates are locked up in solitary confinement, where they must be isolated from the world for six months before they\\'re released onto the general jail population to participate in other inmates. Others are housed in "theforgotten floor," the top floor of the pretrial facility that is also used by criminal offenders. What makes these inmates separate is their medical needs. They\\'re given special medications to treat their mental illness, but it\\'s only in those cases where necessary to take certain medications themselves for the very dangerous condition. These medications can lead to anaphylaxis or dangerous reactions if taken by people who are on them alone. A mentally ill person like John Brown, a convicted felon with severe mental illness. He spends four months in the "forgotten floor" and is charged with being a felon, disorderly conduct, assault with a deadly weapon (a gun), possession of child pornography, resisting arrest, and posses...

[ ] summaries['gpt2'] = "\\n".join(sent_tokenize(pipe_out[0][\"generated_text\"][:len(gpt2_query) :]))

```

Fig : performing summarization
(Source: Self-developed)

STEP 6:

- Now we will be using another model called T5 which is a text- to- transfer-transformer is a pre-trained text summarization model. It takes input and returns modified output text as the output .
- This is better than BERT because bert only outputs a class of texts. T5 model is also used in performing various other NLP tasks like question-answering, machine translation and other classification tasks

```

#downloading the T-5 model and adding the model to the pipeline.

pipe = pipeline('summarization', model = 't5-small' )

#Printing the sample text data in the dataset
pipe_out = pipe(sample_text)

Downloading: 100% 1.20k/1.20k [00:00<00:00, 40.6kB/s]
Downloading: 100% 242M/242M [00:04<00:00, 22.1MB/s]
Downloading: 100% 792k/792k [00:01<00:00, 414kB/s]
Downloading: 100% 1.39M/1.39M [00:01<00:00, 972kB/s]
/usr/local/lib/python3.8/dist-packages/transformers/models/t5/tokenization_t5_fast.py:156: FutureWarning: This tokenizer was incorrectly instantiated with a model path of t5-small. For now, this behavior is kept to avoid breaking backwards compatibility when padding/encoding with "truncation is True".
- Be aware that you SHOULD NOT rely on t5-small automatically truncating your input to 512 when padding/encoding.
- If you want to encode/pad to sequences longer than 512 you can either instantiate this tokenizer with "model_max_length" or pass "max_length" when encoding/padding.
- To avoid this warning, please instantiate this tokenizer with "model_max_length" set to your preferred value.
warnings.warn(

[ ] #Printing the summarized text
pipe_out

[{'summary_text': "inmates with the most severe mental illnesses are incarcerated until they're ready to appear in court . most often, they face drug charges or charges of assaulting an officer . mentally ill people become more paranoid, delusional, and less likely to follow dir ."}]

[ ] summaries['t5'] = '\n'.join(sent_tokenize(pipe_out[0]['summary_text']))

```

Fig : Implementing T5 model

(Source: Self-developed)

STEP 7:

- Performing text summarization using BART.
- BART is an autoencoder for pre-training sequence to sequence model it is trained by Corrupting text with a noise function.Learning a model to construct the original text again,it uses a sequence to sequence model architecture with a bi-directional encoder,
- We will be first adding this pre-trained model into our pipeline and and printing the summarized text on which is being performed on our dataset then joining the summarized text into the dictionary which we created at the beginning.

```

# Addinh the model to the pipeline
pipe = pipeline("summarization", model="facebook/bart-large-cnn")
pipe_out = pipe(sample_text)

Downloading: 100% 1.58k/1.58k [00:00<00:00, 32.1kB/s]
Downloading: 100% 1.63G/1.63G [00:33<00:00, 68.7MB/s]
Downloading: 100% 899k/899k [00:01<00:00, 877kB/s]
Downloading: 100% 456k/456k [00:01<00:00, 574kB/s]
Downloading: 100% 1.36M/1.36M [00:01<00:00, 882kB/s]

[ ] # printing the summarized text
pipe_out

[{'summary_text': 'Miami-Dade pretrial detention facility is dubbed the "forgotten floor" Here, inmates with the most severe mental illnesses are incarcerated. Most often, they face drug charges or charges of assaulting an officer. Judge Steven Leifman says the arrests often result from confrontations with police.'}]

[ ] summaries["bart"] = "\n".join(sent_tokenize(pipe_out[0]['summary_text']))

[ ] summaries["bart"]

'Miami-Dade pretrial detention facility is dubbed the "forgotten floor" Here, inmates with the most severe mental illnesses are incarcerated.\nMost often, they face drug charges or charges of assaulting an officer.\nJudge Steven Leifman says the arrests often result from confrontations with police.'

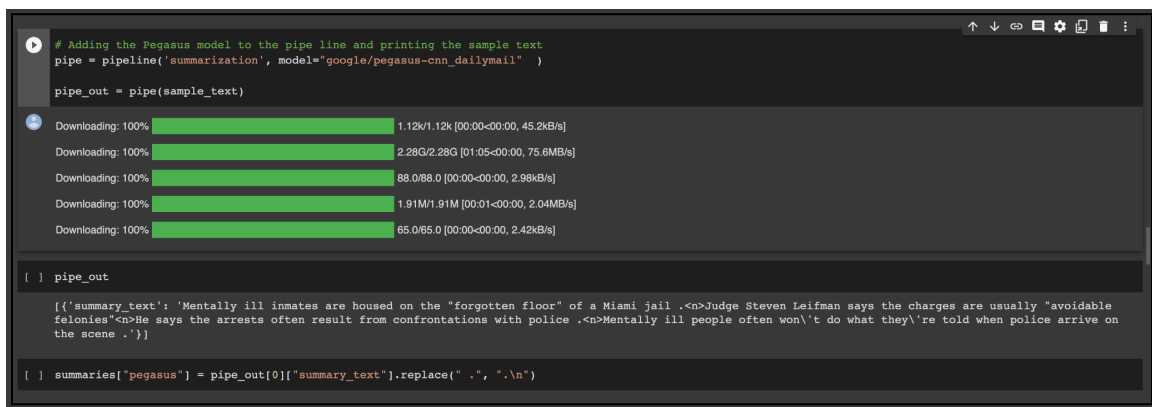
```

Fig : Applying BART

(Source: Self-developed)

STEP 8:

- Here we will be implementing pegasus which is a pre-trained model developed by google.
- This is also an encoder-decoder based model developed by google for sequence to sequence learning.
- Here, the important sentences are removed and marked from an input and later generated together as an output sequence from remaining sentences.
- Here we will add the pegasus model into the pipeline and generate the summarized text from our test dataset and add it to the summaries dictionary.



```
# Adding the Pegasus model to the pipe line and printing the sample text
pipe = pipeline('summarization', model="google/pegasus-cnn_dailymail" )

pipe_out = pipe(sample_text)
```

Downloading: 100% 1.12k/1.12k [00:00<00:00, 45.2kB/s]

Downloading: 100% 2.28G/2.28G [01:05<00:00, 75.6MB/s]

Downloading: 100% 88.0/88.0 [00:00<00:00, 2.98kB/s]

Downloading: 100% 1.91M/1.91M [00:01<00:00, 2.04MB/s]

Downloading: 100% 65.0/65.0 [00:00<00:00, 2.42kB/s]

```
[ ] pipe_out
```

```
{'summary_text': 'Mentally ill inmates are housed on the "forgotten floor" of a Miami jail .<n>Judge Steven Leifman says the charges are usually "avoidable felonies"<n>He says the arrests often result from confrontations with police .<n>Mentally ill people often won\'t do what they\'re told when police arrive on the scene .'}}
```

```
[ ] summaries["pegasus"] = pipe_out[0]["summary_text"].replace(" .", ".\n")
```

Fig : IMplementing Pegasus

(Source: Self-developed)

STEP 9:

- Now we will compare all the summaries we got from the different models, which we have added to the dictionary .

```

# Here we will be comparing the different pretrained models that
# we added to the pipeline and their outputs
print("GROUND TRUTH")

print(dataset['train'][1]['highlights'])

for model_name in summaries:
    print(model_name.upper())
    print(summaries[model_name])

```

GROUND TRUTH

Mentally ill inmates in Miami are housed on the "forgotten floor"

Judge Steven Leifman says most are there as a result of "avoidable felonies"

While CNN tours facility, patient shouts: "I am the son of the president"

Leifman says the system is unjust and he's fighting for change .

BASELINE

Editor's note: In our Behind the Scenes series, CNN correspondents share their experiences in covering news and analyze the stories behind the events.

Here, Soledad O'Brien takes users inside a jail where many of the inmates are mentally ill. An inmate housed on the "forgotten floor," where many mentally ill is

MIAMI, Florida (CNN) -- The ninth floor of the Miami-Dade pretrial detention facility is dubbed the "forgotten floor."

GPT2

MIAMI-DADE, Florida | April 13, 2012 -- Some inmates are locked up in solitary confinement, where they must be isolated from the world for six months before they

Others are housed in "theforgotten floor," the top floor of the pretrial facility that is also used by criminal offenders.

What makes these inmates separate is their medical needs.

They're given special medications to treat their mental illness, but it's only in those cases where necessary to take certain medications themselves for the very

These medications can lead to anaphylaxis or dangerous reactions if taken by people who are on them alone.

A mentally ill person like John Brown, a convicted felon with severe mental illness.

He spends four months in the "forgotten floor" and is charged with being a felon, disorderly conduct, assault with a deadly weapon (a gun), possession of child

Brown was in the second floor when he fell asleep in bed.

On April 11, his bed fell over and he was arrested and held until jail officials could check on him.

After his detention, Brown refused to return to court the next morning to answer the charges.

"I won't be back at all," Brown said.

"There will be no new hearings or there will be no court."

This episode will air on January 18, 2013 on CNN's Inside Story.

T5

Inmates with the most severe mental illnesses are incarcerated until they're ready to appear in court .most often, they face drug charges or charges of assault:

BART

Miami-Dade pretrial detention facility is dubbed the "forgotten floor" Here, inmates with the most severe mental illnesses are incarcerated.

Fig : Comparing summaries

(Source: Self-developed)

STEP 10:

- Now we will calculate the performance metric for the summaries which were generated by different models.
- we have used ScareBLEU which is a part of BLEU score We have calculated precision, recall and other performance metrics.
- BLEU is used to measure the precision of the model whereas ROUGE is used to measure the recall of the model

```

#Downloading the SacreBleu metric and loading the metric in a variable.
from datasets import load_metric

bleu_metric = load_metric("sacrebleu")

<ipython-input-23-215de13eac3>:14: FutureWarning: load_metric is deprecated and will be removed in the next major version of datasets. Use 'evaluate.load' in
bleu_metric = load_metric("sacrebleu")
Downloading builder script: 7.65M? [00:00<00:00, 276kB/s]

[ ] # Predicting the scores for pegasus model
bleu_metric.add(prediction = [summaries["pegasus"]], reference = [dataset['train'][1]['highlights'] ])

results = bleu_metric.compute(smooth_method = 'floor', smooth_value = 0 )

results['precision'] = [np.round(p , 2) for p in results['precisions']]

pd.DataFrame.from_dict(results, orient = 'index', columns = ['Value'])

```

	Value
score	18.456308
counts	[29, 16, 11, 7]
totals	[76, 75, 74, 73]
precisions	[38.1578947368421, 21.333333333333332, 14.8648...
bp	1.0
sys_len	76
ref_len	57
precision	[38.16, 21.33, 14.86, 9.59]

Fig : Calculating performance metrics
(Source: Self-developed)

```

#Downloading the Rouge metric and loading the metric in a variable.
rouge_metric = load_metric('rouge')

Downloading builder script: 5.60K? [00:00<00:00, 183kB/s]

[ ] # Predicting the rouge scores for all the models
rouge_names = ["rouge1", "rouge2", "rougeL", "rougeLsum"]

reference = dataset['train'][1]['highlights']

records = []

for model_name in summaries:
    rouge_metric.add(prediction = summaries[model_name], reference = reference )
    score = rouge_metric.compute()
    rouge_dict = dict((rn, score[rn].mid.fmeasure) for rn in rouge_names )
    print('rouge_dict ', rouge_dict )
    records.append(rouge_dict)

pd.DataFrame.from_records(records, index = summaries.keys() )

```

```

rouge_dict {'rouge1': 0.365079365079365, 'rouge2': 0.14516129032258066, 'rougeL': 0.20634920634920634, 'rougeLsum': 0.2857142857142857}
rouge_dict {'rouge1': 0.1836734693877551, 'rouge2': 0.0410958904109589, 'rougeL': 0.10204081632653061, 'rougeLsum': 0.17006802721088435}
rouge_dict {'rouge1': 0.1758241758241758, 'rouge2': 0.0, 'rougeL': 0.13186813186813187, 'rougeLsum': 0.15384615384615383}
rouge_dict {'rouge1': 0.3655913978494624, 'rouge2': 0.13186813186813184, 'rougeL': 0.2150537634408602, 'rougeLsum': 0.3225806451612903}
rouge_dict {'rouge1': 0.49019607843137253, 'rouge2': 0.24000000000000002, 'rougeL': 0.3529411764705882, 'rougeLsum': 0.4509803921568628}

```

Fig : Calculating performance metrics
(Source: Self-developed)

RESULTS:

- From the above we we have seen how each state of the art model has generated summaries for the text in our dataset and we have seen how the summaries were now we have evaluated.
- We have performed ROUGE scores for all the models and came up with the following result.

	rouge1	rouge2	rougeL	rougeLsum
baseline	0.365079	0.145161	0.206349	0.285714
gpt2	0.183673	0.041096	0.102041	0.170068
t5	0.175824	0.000000	0.131868	0.153846
bart	0.365591	0.131868	0.215054	0.322581
pegasus	0.490196	0.240000	0.352941	0.450980

Fig : showing results

(Source: Self-developed)

Project management

Work completed:

Description

Collected the dataset from Kaggle platform and applied preprocessing. Performed data analysis on the dataset to understand the nature and structure of the dataset. Implemented multiple models and developed models to perform summarization. First we implement the Transformer model that uses Encoder and Decoder approach to summarize. We also developed a model using GPT-2 which is a unsupervised deep learning model by Hugging Face module. We also

developed Bert model which is a classification based model. Then we also developed BART model which uses a Sequence-to-Sequence model which makes use of Bi-directional encoder for summarization. And for our last model we implement a model using Pegasus which is a pretrained model developed by Google. Using this model reduces training time and computing resources while yielding generalised results. Computed evaluation metrics on each model to obtain accuracy and other performance metrics and compared each model's performance.

Responsibility :

Manish Reddy Radha Reddy : Worked on increasing the accuracy of the model developed in increment 1. Developed the main state of the art model for summarization.

Ramya Sree Konduru : Performed analysis on the dataset and transformed the dataset to better fit models. Evaluated other existing models and analysed its weakness.

Hari Vamshi Krishna : Involved in building multiple models and performing summarization. Performed model training and validation. Tested multiple model performance changes by tweaking model metrics.

Ranjith Kumar Sadafule : Involved in building multiple models and performing summarization. Computed performance metrics and compared each model's performance.

Contribution:

Manish Reddy Radha Reddy : 25%

Ramya Sree Konduru : 25%

Hari Vamsi Krishna Samayamantri :25%

Ranjith Kumar Sadafule :25%

References

- Anand, C., 2021. Comparison of stock price prediction models using pre-trained neural networks. *Journal of Ubiquitous Computing and Communication Technologies (UCCT)*, 3(02), pp.122-134.
- Hu, Z., Zhao, Y. and Khushi, M., 2021. A survey of forex and stock price prediction using deep learning. *Applied System Innovation*, 4(1), p.9.
- Mehtab, S. and Sen, J., 2020. Stock price prediction using convolutional neural networks on a multivariate timeseries. *arXiv preprint arXiv:2001.09769*.
- Obthong, M., Tantisantiwong, N., Jeamwatthanachai, W. and Wills, G., 2020. A survey on machine learning for stock price prediction: algorithms and techniques.