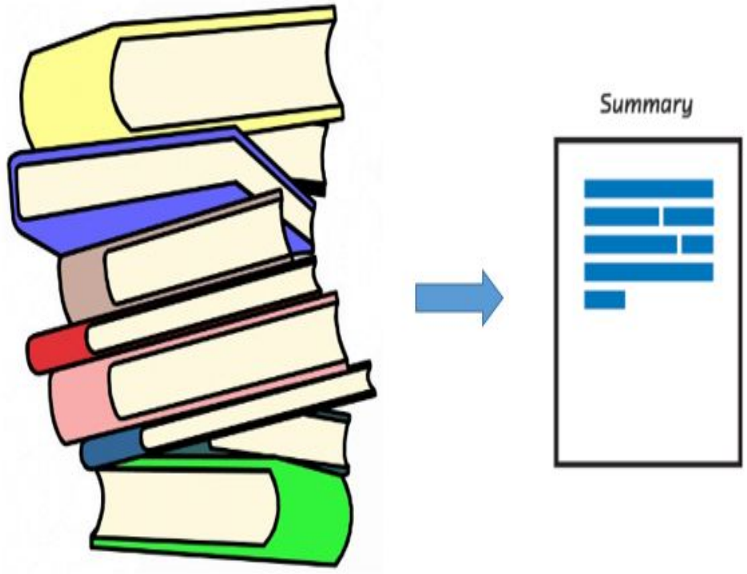
TEXT SUMMARIZATION

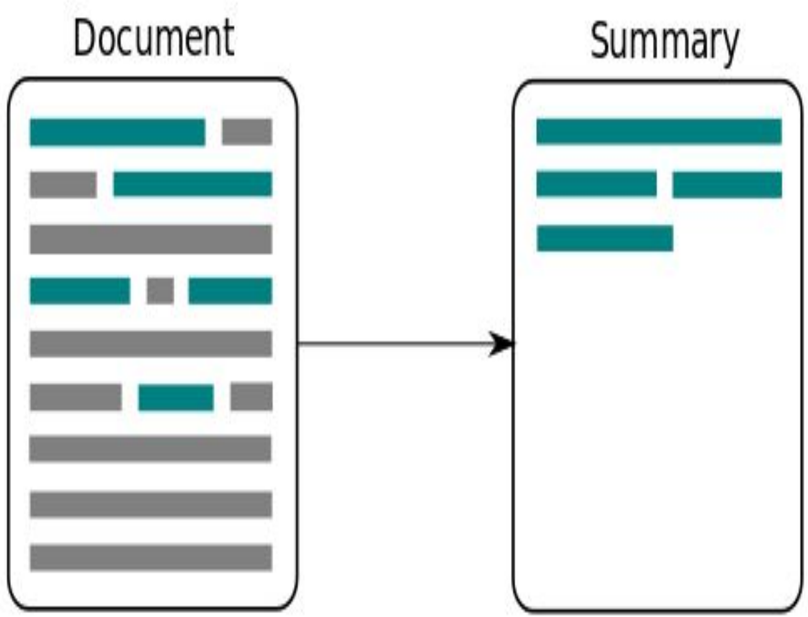


MEMBERS
REENA YADAV(Rollno-33)
ROHINI SINGH(Rollno-36)
SAUMYA GUPTA(Rollno-39)
VIKAS(Rollno-55)

What is Text summarization ?

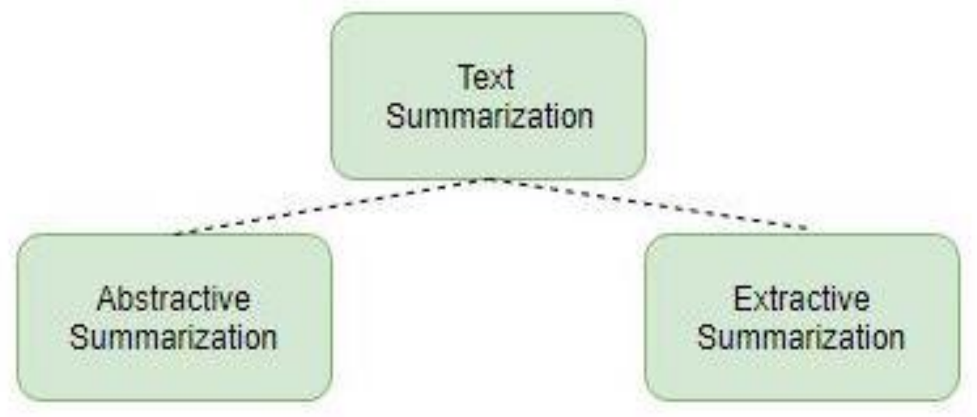
Text summarization refers to the technique of shortening long pieces of text or The technique, where a computer program shortens longer texts and generates summaries to pass the intended message.

The method of extracting these summaries from the original huge text without losing vital information is called as Text Summarization. It is essential for the summary to be a fluent, continuous and depict the significant.



Types of text summarization

There are broadly two different approaches that are used for text summarization:



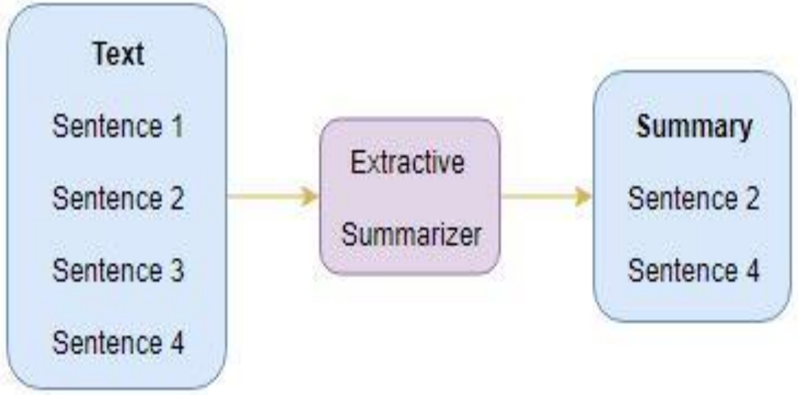
1.Extractive Summarization

It is the traditional method developed first. The main objective is to identify the significant sentences of the text and add them to the summary. You need to note that the summary obtained contains exact sentences from the original text.Extractive strategies are set up as binary classification problems where the goal is to identify the article sentences belonging to the summary.

Here is an example:

Source text: Joseph and Mary rode on a donkey to attend the annual event in Jerusalem.The event was very crowded.In the city, Mary gave birth to a child named Jesus.

Extractive summary: Joseph and Mary rode on a donkey to attend the annual event in Jerusalem.In the city, Mary gave birth to a child named Jesus.

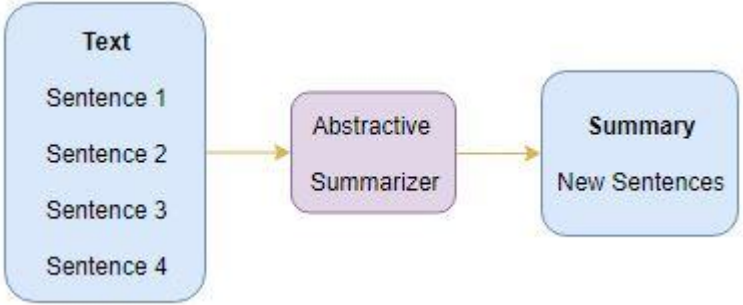


2.Abstractive Summarization

It is a more advanced method, the approach is to identify the important sections, interpret the context and reproduce in a new way. This ensures that the core information is conveyed through shortest text possible.Abstractive summaries need to identify the key points and then add a generative element. Finally, mixed strategies need to combine these elements and provide a mechanism to decide when each mode should be used.Here is an example:

Source text: Joseph and Mary rode on a donkey to attend the annual event in Jerusalem. In the city, Mary gave birth to a child named Jesus.

Abstractive summary: Joseph and Mary came to Jerusalem where Jesus was born.



Text

Sentence 1

Sentence 2

Sentence 3

Sentence 4

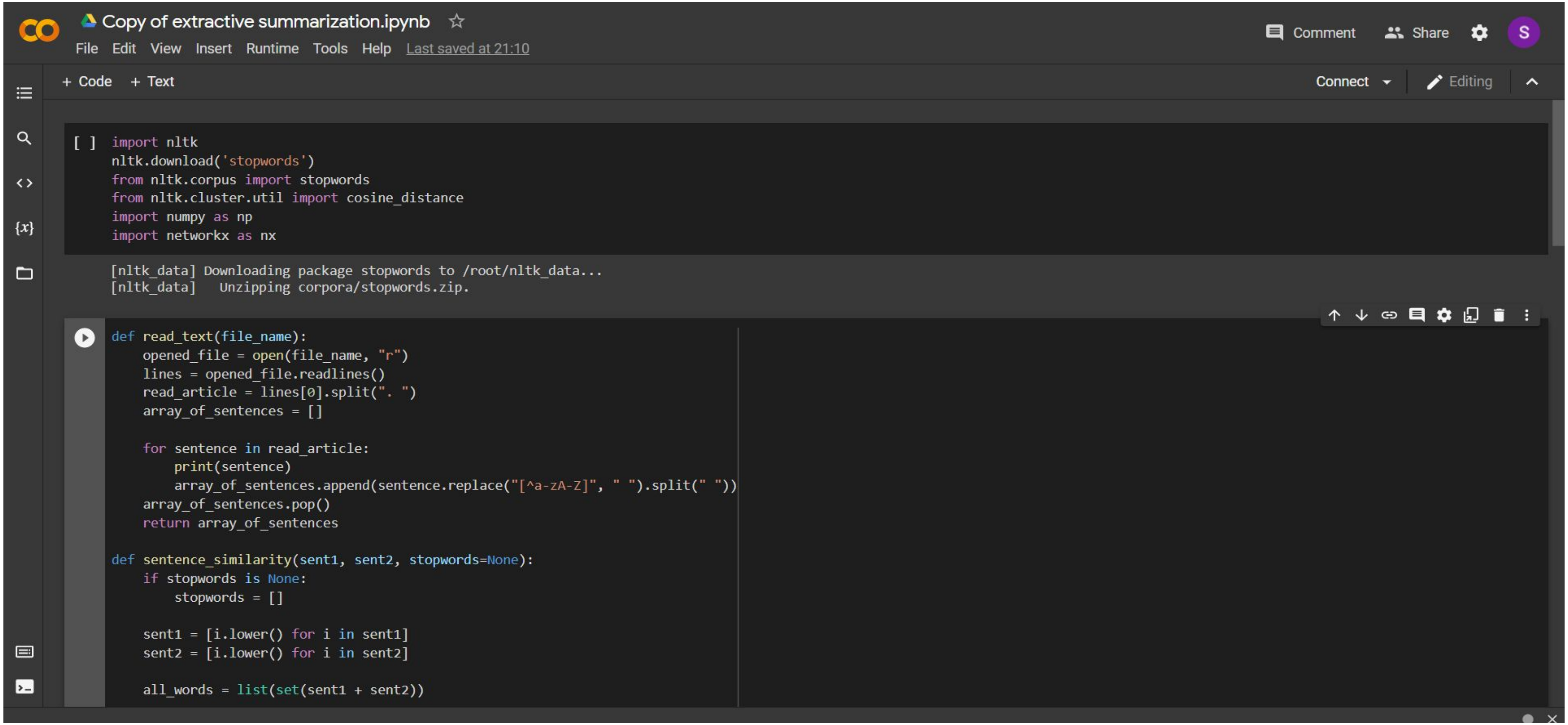
Abstractive Summarizer

Summary

New Sentences

IMPLEMENTATION OF EXTRACTIVE SUMMARIZATION USING NLTK LIBRARY

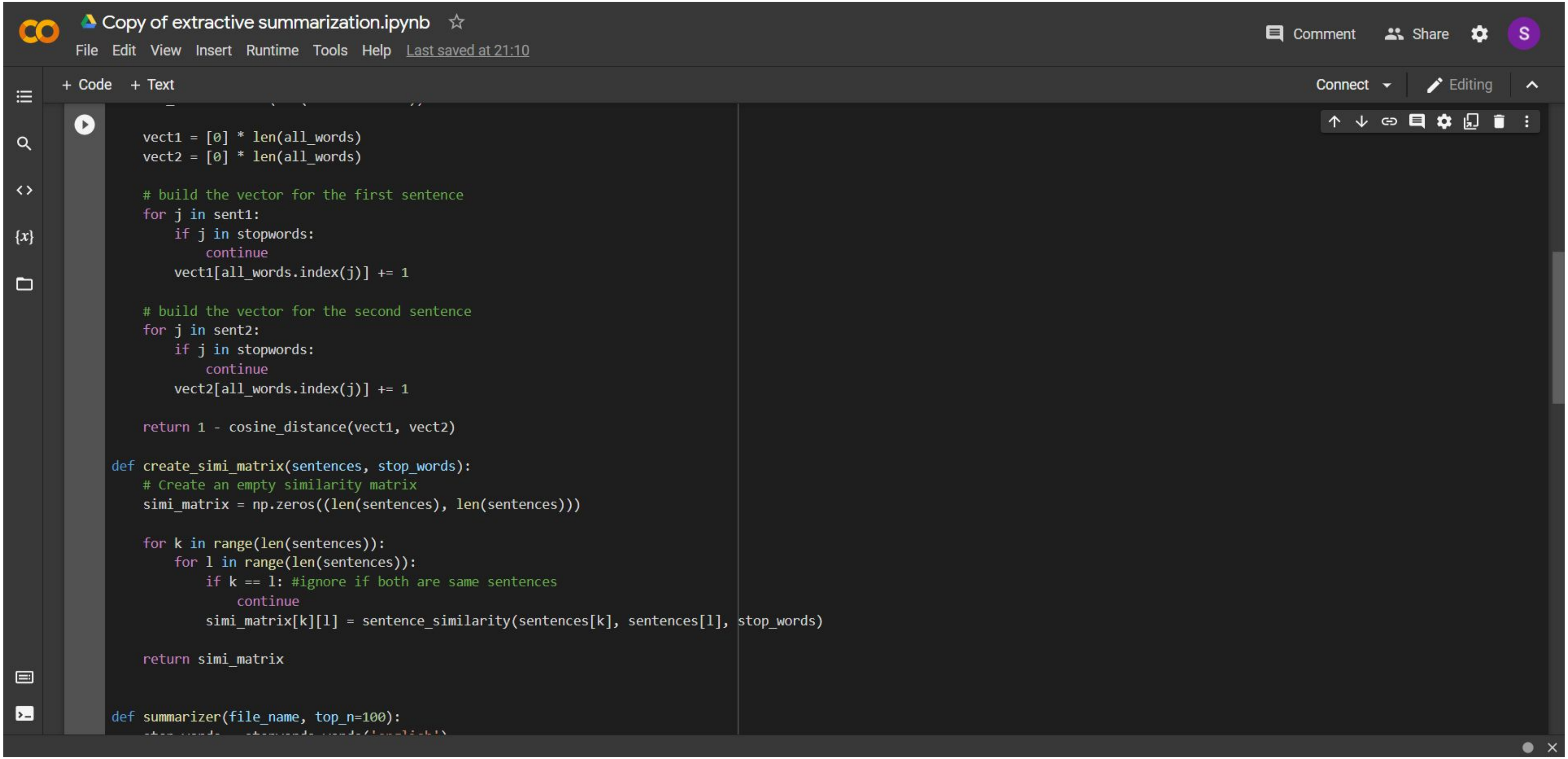
Extractive summarization


Copy of extractive summarization.ipynb
File Edit View Insert Runtime Tools Help Last saved at 21:10
Comment Share S
+ Code + Text
Connect
Editing

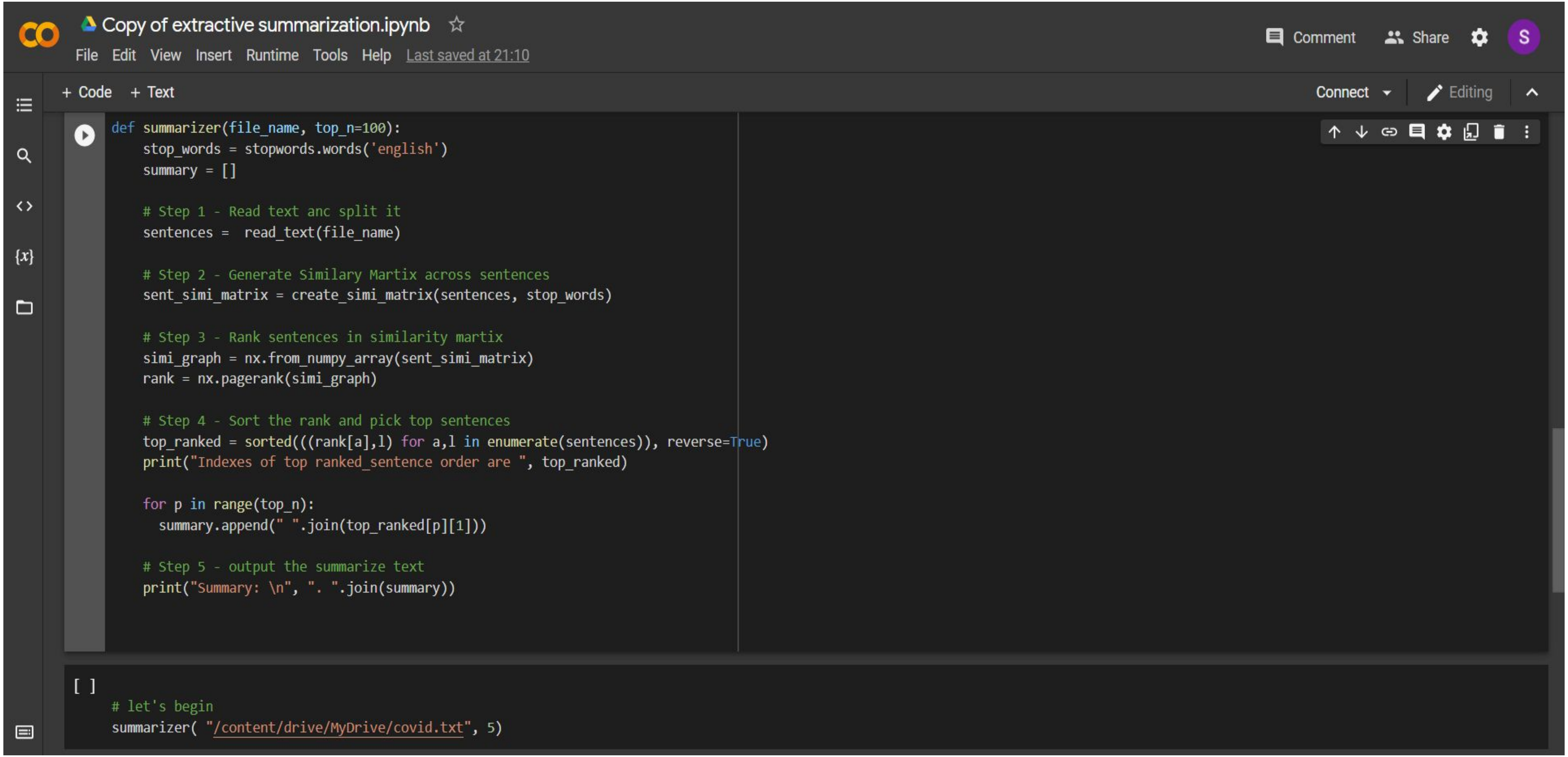
```
[ ] import nltk
    nltk.download('stopwords')
    from nltk.corpus import stopwords
    from nltk.cluster.util import cosine_distance
    import numpy as np
    import networkx as nx
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
def read_text(file_name):
    opened_file = open(file_name, "r")
    lines = opened_file.readlines()
    read_article = lines[0].split(". ")
    array_of_sentences = []

    for sentence in read_article:
        print(sentence)
        array_of_sentences.append(sentence.replace("[^a-zA-Z]", " ").split(" "))
    array_of_sentences.pop()
    return array_of_sentences

def sentence_similarity(sent1, sent2, stopwords=None):
    if stopwords is None:
        stopwords = []

    sent1 = [i.lower() for i in sent1]
    sent2 = [i.lower() for i in sent2]

    all_words = list(set(sent1 + sent2))
```



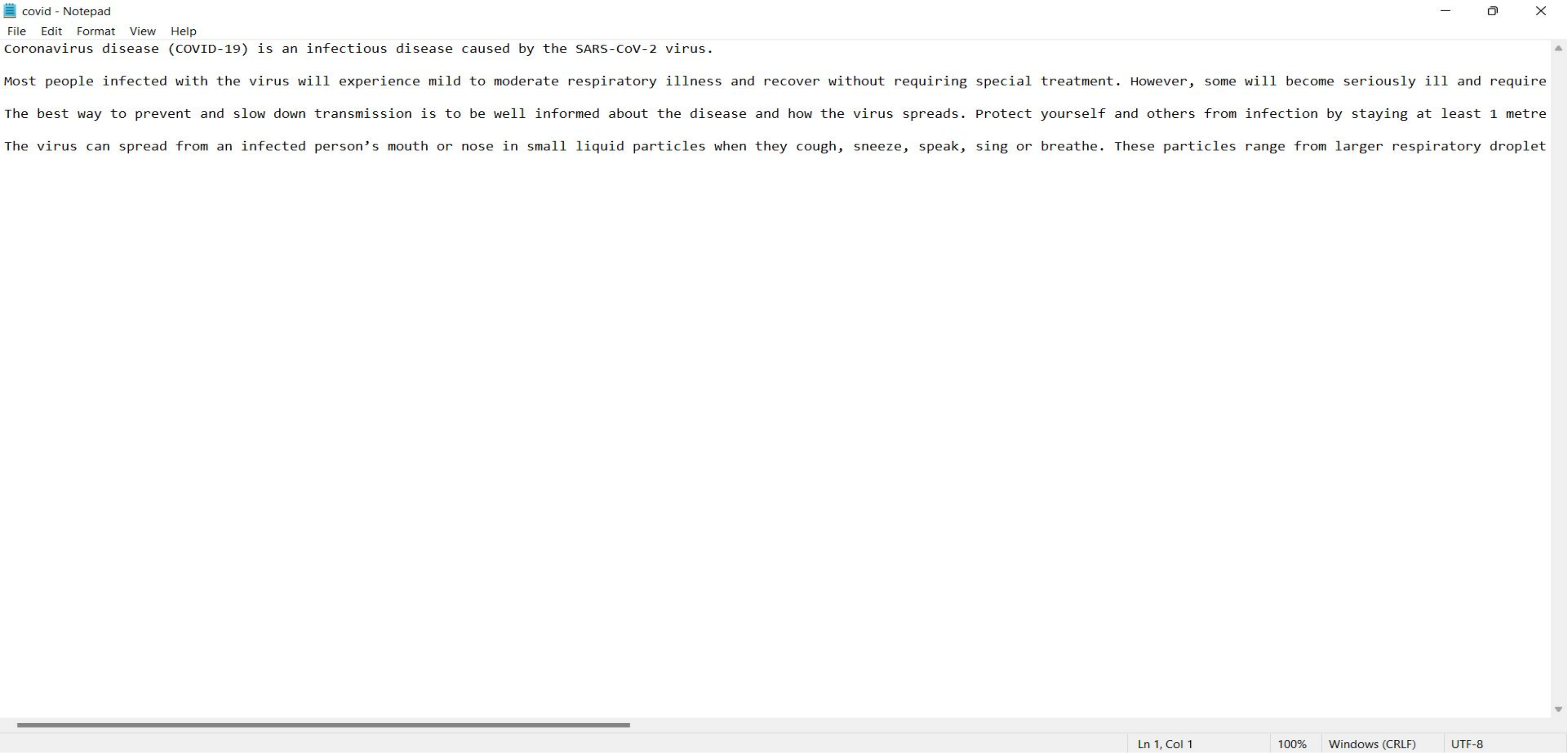
```python
    vect1 = [0] * len(all_words)
    vect2 = [0] * len(all_words)

    # build the vector for the first sentence
    for j in sent1:
        if j in stopwords:
            continue
        vect1[all_words.index(j)] += 1

    # build the vector for the second sentence
    for j in sent2:
        if j in stopwords:
            continue
        vect2[all_words.index(j)] += 1

    return 1 - cosine_distance(vect1, vect2)

def create_simi_matrix(sentences, stop_words):
    # Create an empty similarity matrix
    simi_matrix = np.zeros((len(sentences), len(sentences)))

    for k in range(len(sentences)):
        for l in range(len(sentences)):
            if k == l: #ignore if both are same sentences
                continue
            simi_matrix[k][l] = sentence_similarity(sentences[k], sentences[l], stop_words)

    return simi_matrix


def summarizer(file_name, top_n=100):
    stop_words = stopwords.words('english')
```



```
def summarizer(file_name, top_n=100):
    stop_words = stopwords.words('english')
    summary = []

    # Step 1 - Read text anc split it
    sentences =  read_text(file_name)

    # Step 2 - Generate Similary Martix across sentences
    sent_simi_matrix = create_simi_matrix(sentences, stop_words)

    # Step 3 - Rank sentences in similarity martix
    simi_graph = nx.from_numpy_array(sent_simi_matrix)
    rank = nx.pagerank(simi_graph)

    # Step 4 - Sort the rank and pick top sentences
    top_ranked = sorted(((rank[a],l) for a,l in enumerate(sentences)), reverse=True)
    print("Indexes of top ranked_sentence order are ", top_ranked)

    for p in range(top_n):
      summary.append(" ".join(top_ranked[p][1]))

    # Step 5 - output the summarize text
    print("Summary: \n", ". ".join(summary))
```

```
[ ]
    # let's begin
    summarizer( "/content/drive/MyDrive/covid.txt", 5)
```

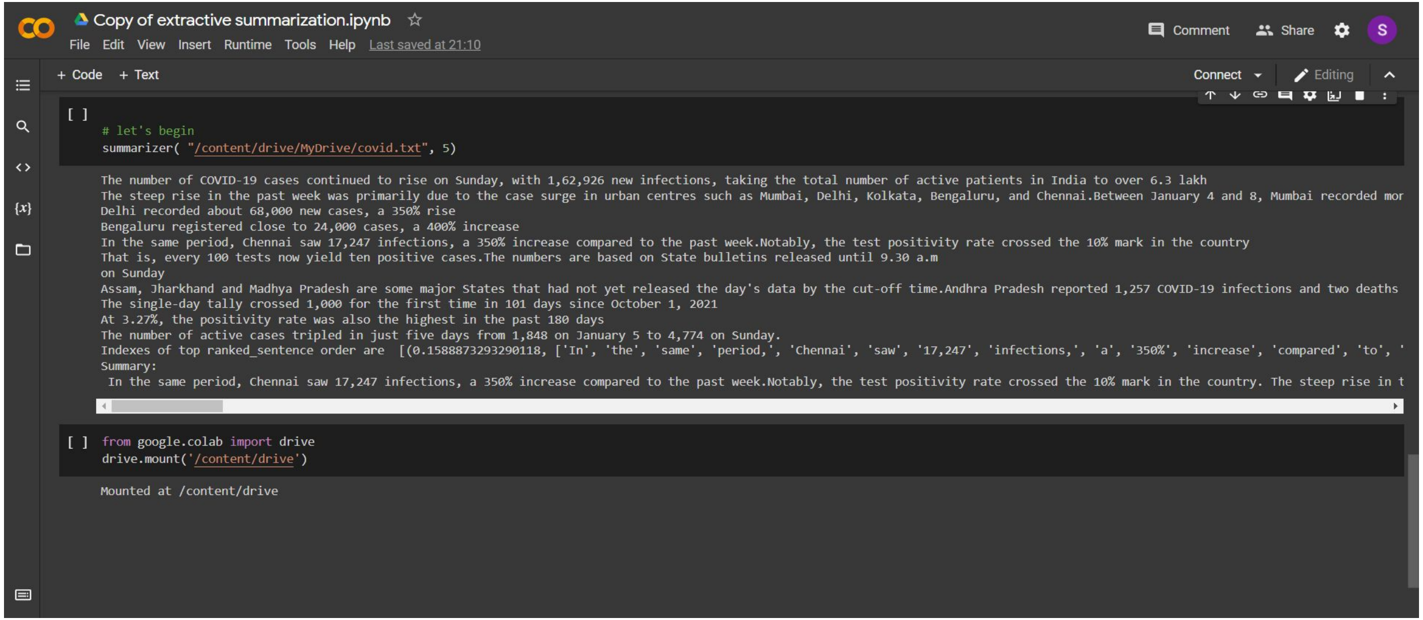
Text file



```
Coronavirus disease (COVID-19) is an infectious disease caused by the SARS-CoV-2 virus.

Most people infected with the virus will experience mild to moderate respiratory illness and recover without requiring special treatment. However, some will become seriously ill and require

The best way to prevent and slow down transmission is to be well informed about the disease and how the virus spreads. Protect yourself and others from infection by staying at least 1 metre

The virus can spread from an infected person’s mouth or nose in small liquid particles when they cough, sneeze, speak, sing or breathe. These particles range from larger respiratory droplet
```

We have used NLTK suite of libraries to implement extractive code summarizer.**NLTK (Natural Language Toolkit)** Library is a suite that contains libraries and programs for symbolic and statistical language processing.

SUMMARY OF THE TEXT


Copy of extractive summarization.ipynb
File Edit View Insert Runtime Tools Help Last saved at 21:10
Comment Share
S
+ Code + Text
Connect
Editing

```
[ ]
    # let's begin
    summarizer( "/content/drive/MyDrive/covid.txt", 5)
```

```
The number of COVID-19 cases continued to rise on Sunday, with 1,62,926 new infections, taking the total number of active patients in India to over 6.3 lakh
The steep rise in the past week was primarily due to the case surge in urban centres such as Mumbai, Delhi, Kolkata, Bengaluru, and Chennai.Between January 4 and 8, Mumbai recorded mor
Delhi recorded about 68,000 new cases, a 350% rise
Bengaluru registered close to 24,000 cases, a 400% increase
In the same period, Chennai saw 17,247 infections, a 350% increase compared to the past week.Notably, the test positivity rate crossed the 10% mark in the country
That is, every 100 tests now yield ten positive cases.The numbers are based on State bulletins released until 9.30 a.m
on Sunday
Assam, Jharkhand and Madhya Pradesh are some major States that had not yet released the day's data by the cut-off time.Andhra Pradesh reported 1,257 COVID-19 infections and two deaths
The single-day tally crossed 1,000 for the first time in 101 days since October 1, 2021
At 3.27%, the positivity rate was also the highest in the past 180 days
The number of active cases tripled in just five days from 1,848 on January 5 to 4,774 on Sunday.
Indexes of top ranked_sentence order are  [(0.1588873293290118, ['In', 'the', 'same', 'period,', 'Chennai', 'saw', '17,247', 'infections,', 'a', '350%', 'increase', 'compared', 'to', '
Summary:
 In the same period, Chennai saw 17,247 infections, a 350% increase compared to the past week.Notably, the test positivity rate crossed the 10% mark in the country. The steep rise in t
```

```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Pros and cons of NLP library

PROS	***CONS***
● It is a very full and well-known library. ● It has many third party extensions and has plenty of approaches to each NLP task. ● Fast sentence tokenization. ● It supports largest number of languages compared to other libraries.	● NLP Library is very complicated to use and is very slow. ● NLTK only splits text by sentences paying no attention to the semantic structure. ● Doesn’t provide neural network models. ● No integrated word vectors.

IMPLEMENTATION OF EXTRACTIVE SUMMARIZATION USING BERT MODEL

BERT MODEL

BERT (Bidirectional transformer)is a transformer used to overcome the limitations of RNN and other neural networks as Long term dependencies. It is a pre-trained model that is naturally bidirectional. This pre-trained model can be tuned easily to perform the NLP tasks as specified. The one major noticable difference between RNN and BERT is the Self attention layer. The model tries to identify the strongest links between the words and thus helps in representation.

Till the date, BERTS are considered as the best available technique to perform the NLP tasks.

Points to a glance:

- BERT models are pre-trained on huge datasets thus no further training is required.

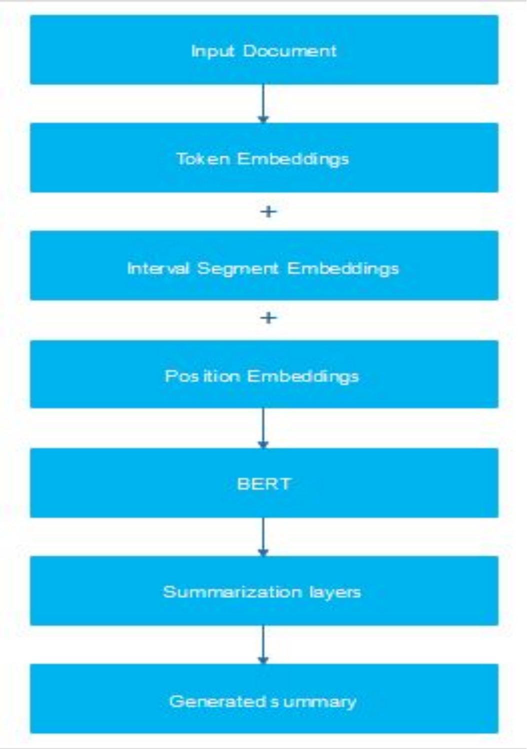
- It uses a powerful flat architecture with inter sentence transform layers so as to get the best results in summarization.

Advantages of BERT model:

- It is most efficient summarizer till date.
- Faster than RNN.
- The Summary sentences are assumed to be representing the most important points of a document.

Methodology

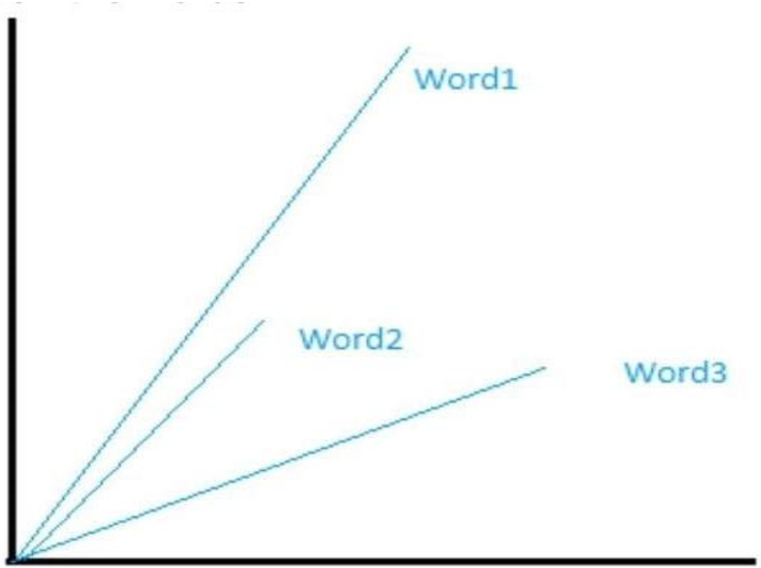
For a set of sentences {sent1,sent2,sent3,...,sentn,} we have two possibilities , that are , yi={0,1} which denotes whether a particular sentence will be picked or not.



Being trained as a masked model the output vectors are tokened instead of sentences. Unlike other extractive summarizers it makes use of embeddings for indicating different sentences and it has only two labels namely sentence A and sentence B rather than multiple sentences. These embeddings are modified accordingly to generate required summaries.

Embeddings

It basically refers to the representation of words in their vector forms. It helps to make their usage flexible. Even the Google utilizes the this feature of BERT for better understanding of queries. It helps in unlocking various functionality towards the semantics from understanding the intent of the document to developing a similarity model between the words.



BERT Architecture

There are following two bert models introduced:

- **BERT base**

In the BERT base model we have 12 transformer layers along with 12 attention layers and 110 million parameters.

- **BERT Large**

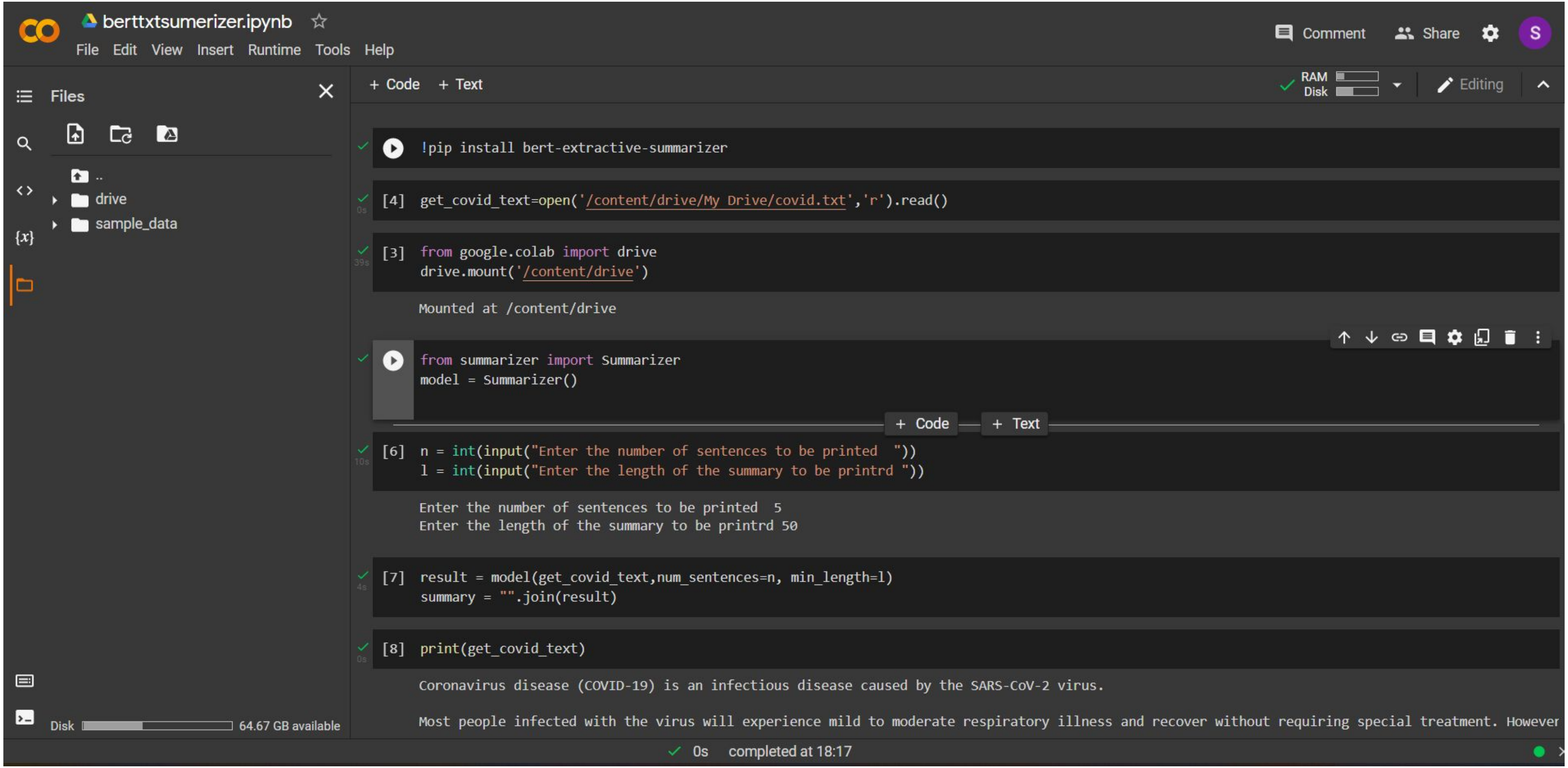
In BERT large model we have 24 transformer layers along with 16 attention layers and 340 million parameters.

Transformer layer- Transformer layer is actually a combination of complete set of encoder and decoder layers and the intermediate connections. Each encoder includes Attention layers along with a RNN. Decoder also has the same architecture but it includes another attention layer in between them as does the seq2seq model. It helps to concentrate on important words.

Summarization layers

We can have different types of layers within the BERT model each having its own specifications:

- **Simple Classifier** - In a simple classifier method , a linear layer is added to the BERT along with a sigmoid function to predict the score Y.
- **Inter Sentence Transformer** - In the inter sentence transformer ,simple classifier is not used. Rather various transformer layers are added into the model only on the sentence representation thus making it more efficient. This helps in recognizing the important points of the document.
- **Recurrent Neural network** - An LSTM layer is added with the BERT model output in order to learn the summarization specific features. Where each LSTM cell is normalized.



```
!pip install bert-extractive-summarizer
```

```
[4] get_covid_text=open('/content/drive/My Drive/covid.txt','r').read()
```

```
[3] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
from summarizer import Summarizer
model = Summarizer()
```

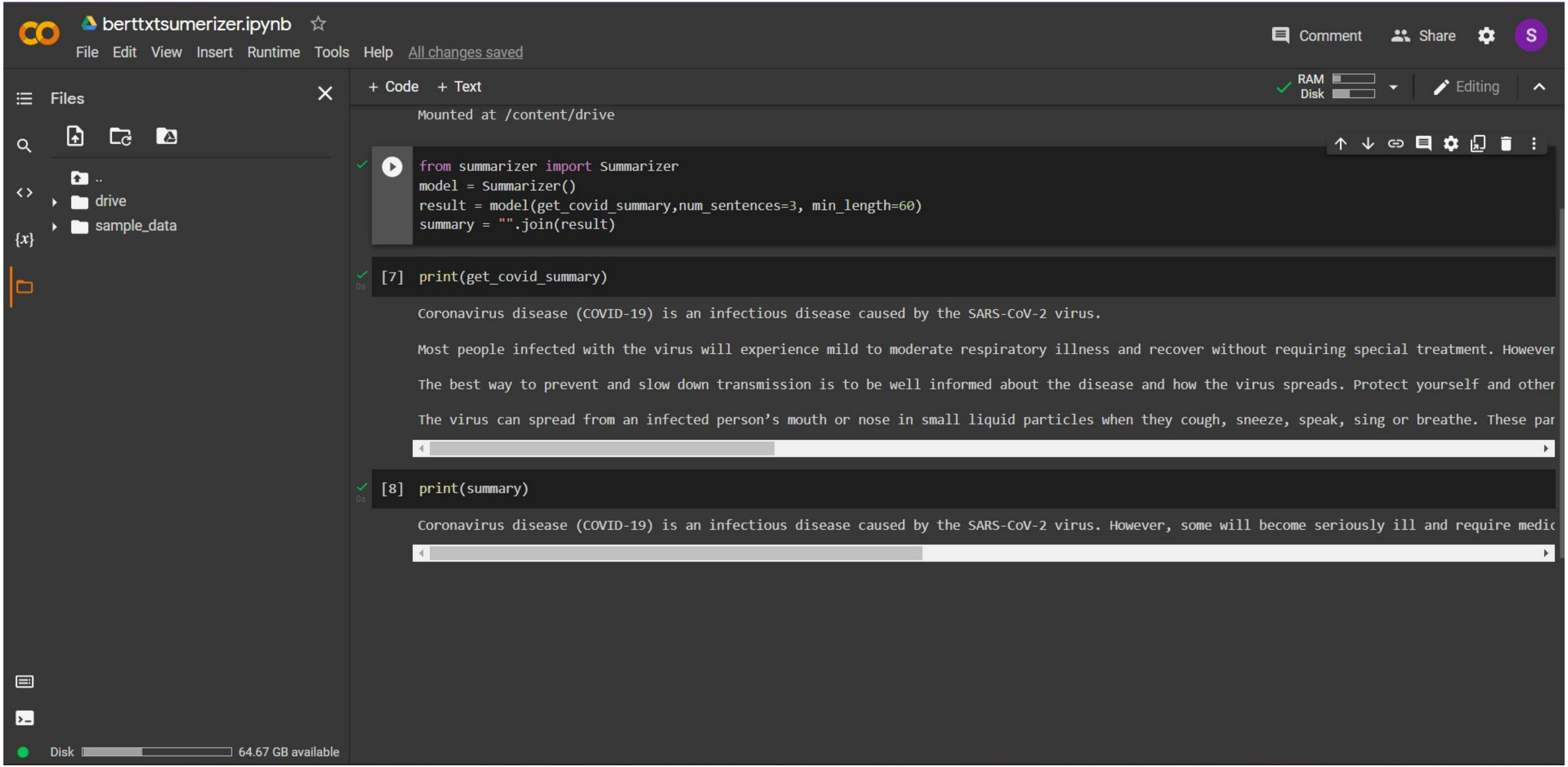
```
[6] n = int(input("Enter the number of sentences to be printed  "))
    l = int(input("Enter the length of the summary to be printrd "))
```

```
Enter the number of sentences to be printed  5
Enter the length of the summary to be printrd 50
```

```
[7] result = model(get_covid_text,num_sentences=n, min_length=l)
    summary = "".join(result)
```

```
[8] print(get_covid_text)
```

```
Coronavirus disease (COVID-19) is an infectious disease caused by the SARS-CoV-2 virus.

Most people infected with the virus will experience mild to moderate respiratory illness and recover without requiring special treatment. However
```



```
Mounted at /content/drive
```

```
from summarizer import Summarizer
model = Summarizer()
result = model(get_covid_summary,num_sentences=3, min_length=60)
summary = "".join(result)
```

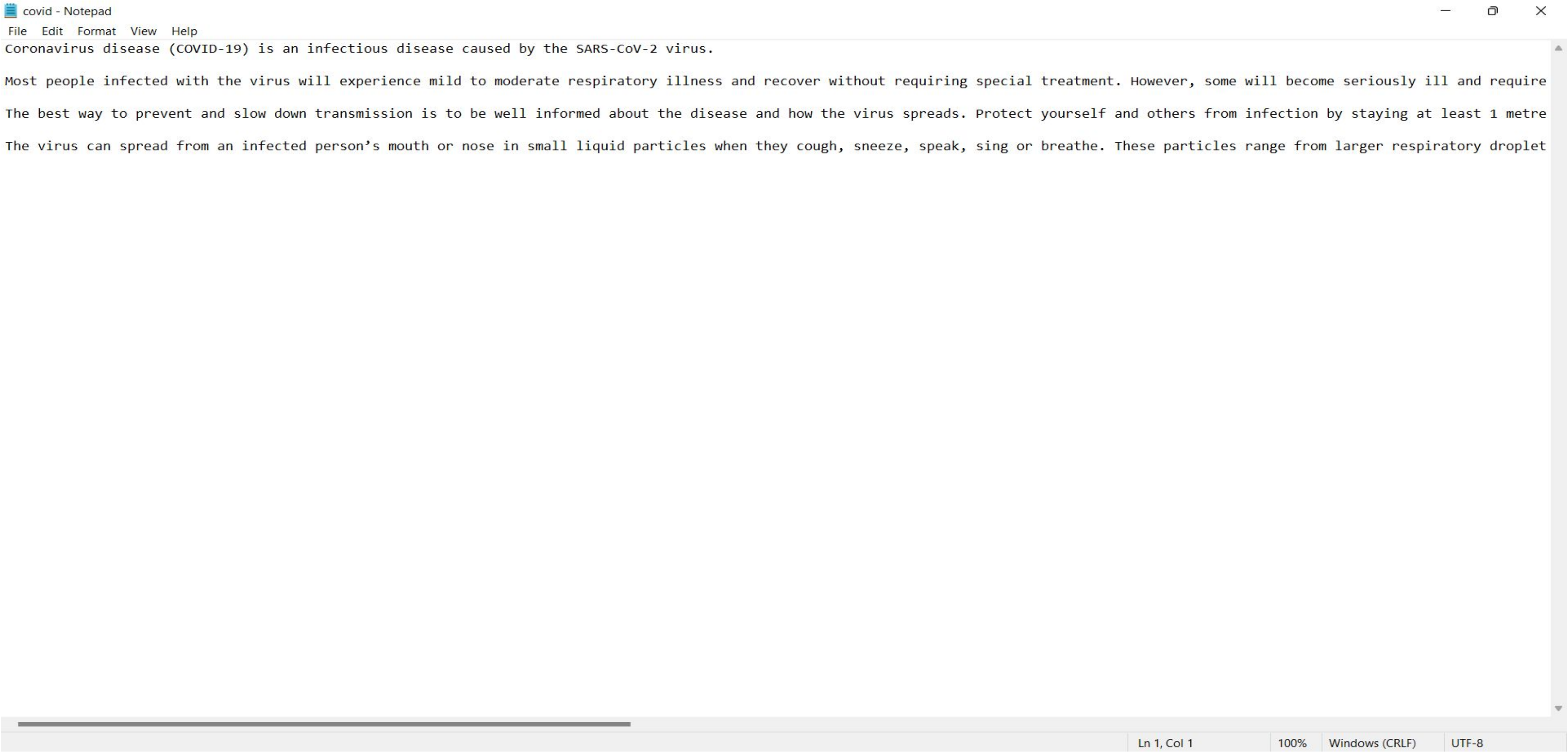
```
[7] print(get_covid_summary)
```

```
Coronavirus disease (COVID-19) is an infectious disease caused by the SARS-CoV-2 virus.

Most people infected with the virus will experience mild to moderate respiratory illness and recover without requiring special treatment. However

The best way to prevent and slow down transmission is to be well informed about the disease and how the virus spreads. Protect yourself and other

The virus can spread from an infected person’s mouth or nose in small liquid particles when they cough, sneeze, speak, sing or breathe. These par
```

```
[8] print(summary)
```

```
Coronavirus disease (COVID-19) is an infectious disease caused by the SARS-CoV-2 virus. However, some will become seriously ill and require medic
```

Text file



```
Coronavirus disease (COVID-19) is an infectious disease caused by the SARS-CoV-2 virus.

Most people infected with the virus will experience mild to moderate respiratory illness and recover without requiring special treatment. However, some will become seriously ill and require

The best way to prevent and slow down transmission is to be well informed about the disease and how the virus spreads. Protect yourself and others from infection by staying at least 1 metre

The virus can spread from an infected person’s mouth or nose in small liquid particles when they cough, sneeze, speak, sing or breathe. These particles range from larger respiratory droplet
```

Need of Text summarization

- Summaries reduce reading time.
- When researching documents, summaries make the selection process easier.
- Automatic summarization improves the effectiveness of indexing.
- Automatic summarization algorithms are less biased than human summarizers.

ABSTRACTIVE SUMMARIZATION

What is a Transformer?

A Transformer is a machine learning architecture that combines an *encoder* with a *decoder* and jointly learns them, allowing us to convert input sequences (e.g. phrases) into some *intermediate format* before we convert it back into human-understandable format.

Transformer have become the primary choice for AI driven language tasks these days because they can apply self-attention and are parallel in nature.

The transformer network is solely based upon multiple attention layers. It does not make use of RNN and is reliant on attention layers and positional encoding for remembering the sequence of words in the input sequence. The global dependencies created with the help of multiple attention layers help in creating parallelization in processing the input.

The transformer model contains encoder and decoder layers, where each is connected to a multi-head attention layer and feed forward network layers. The model remembers the position and sequence of words with the help of cosine and sine functions that creates positional encoding. The multi-head attention layer in the encoder and decoder layer applies a mechanism called self-attention. The input is fed into three connected layers to create query (Q), key (K), and value (V) vectors .These vectors are split into n vectors.

Figure below depicts the architecture of a transformer model. It contains an encoder and decoder layer and the various normalization and multi-head attention layers are also depicted in the figure.

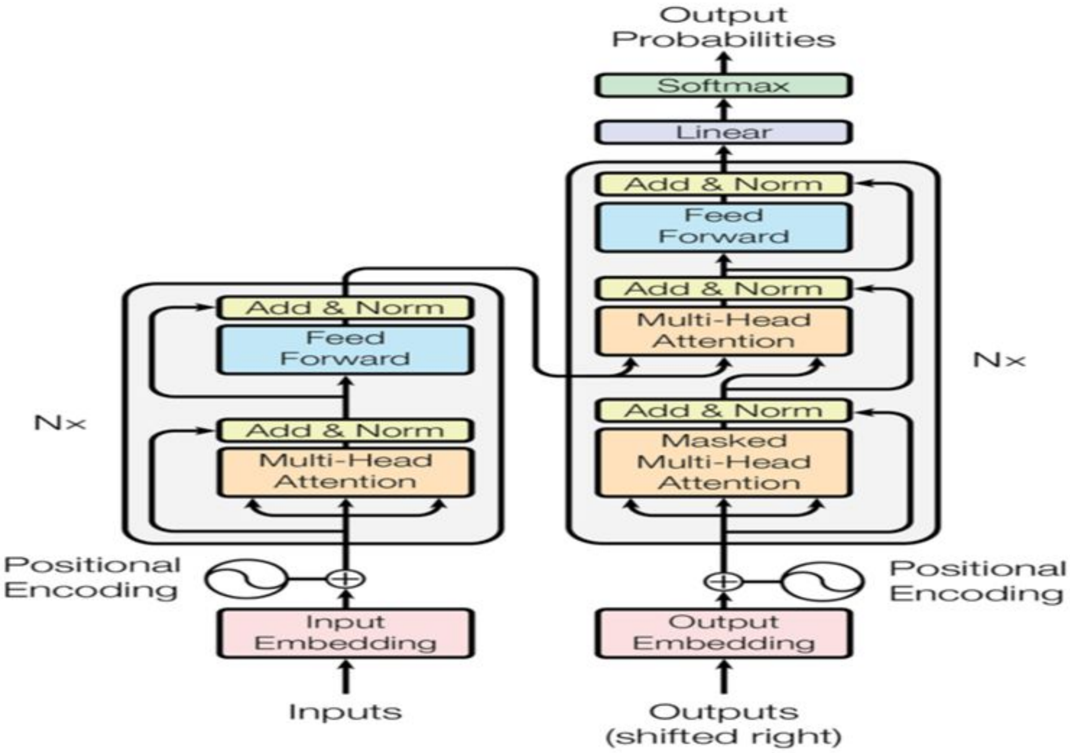


Figure 1: The Transformer - model architecture.

Models that we used based on Transformers - Hugging face works as an opensource for providing many useful NLP libraries and datasets. Its most famous library is the Transformer library. The transformer library consists of various pre-trained models to predict summaries of texts that can be fine-tuned for any dataset. The models we used are as follows:

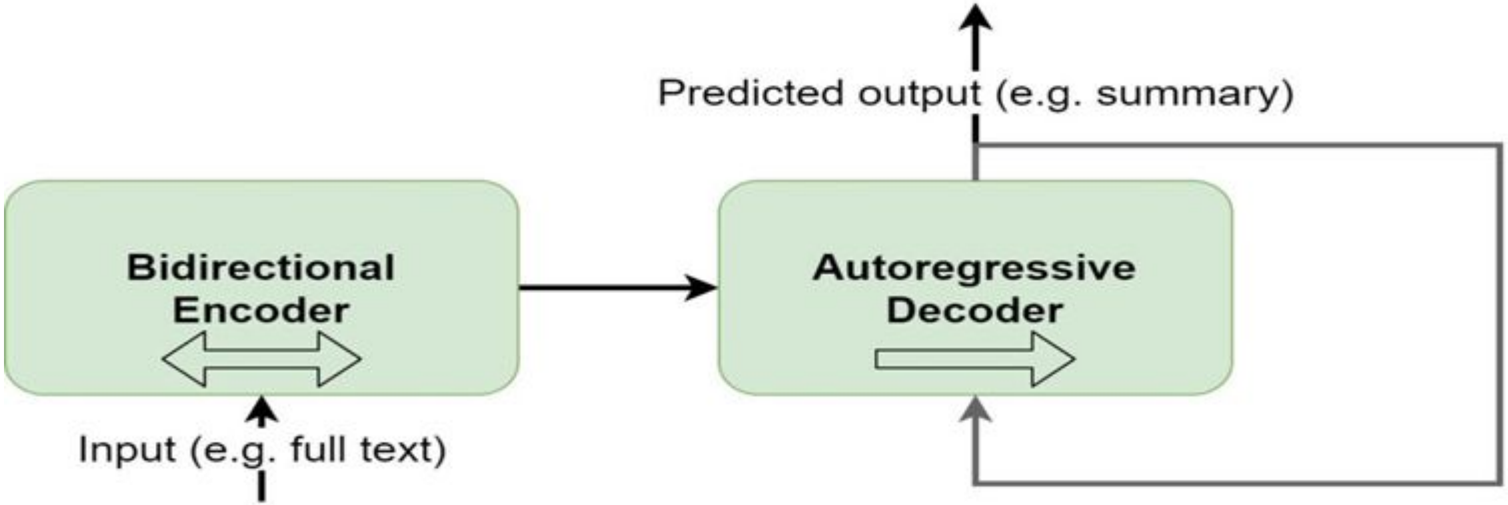
- **Pipeline** – The pipelines are a great and quick way to use different pre-trained models for inference. These pipelines are objects that abstract most of the library's complicated code, offering a simple API dedicated to several tasks, including text summarization. Pipelines enclose the overall steps of every NLP process such as Tokenization, Inference, which maps every token into a more meaningful representation, and Decoding. The Hugging Face transformers summarization pipeline has made the task easier, faster and more efficient to execute in English language. We used the machine learning model that has been trained on the CNN news corpus by using a fine-tuned BART algorithm and is loaded from pipeline() using the task identifier: "summarization".

Now comes what is BART?

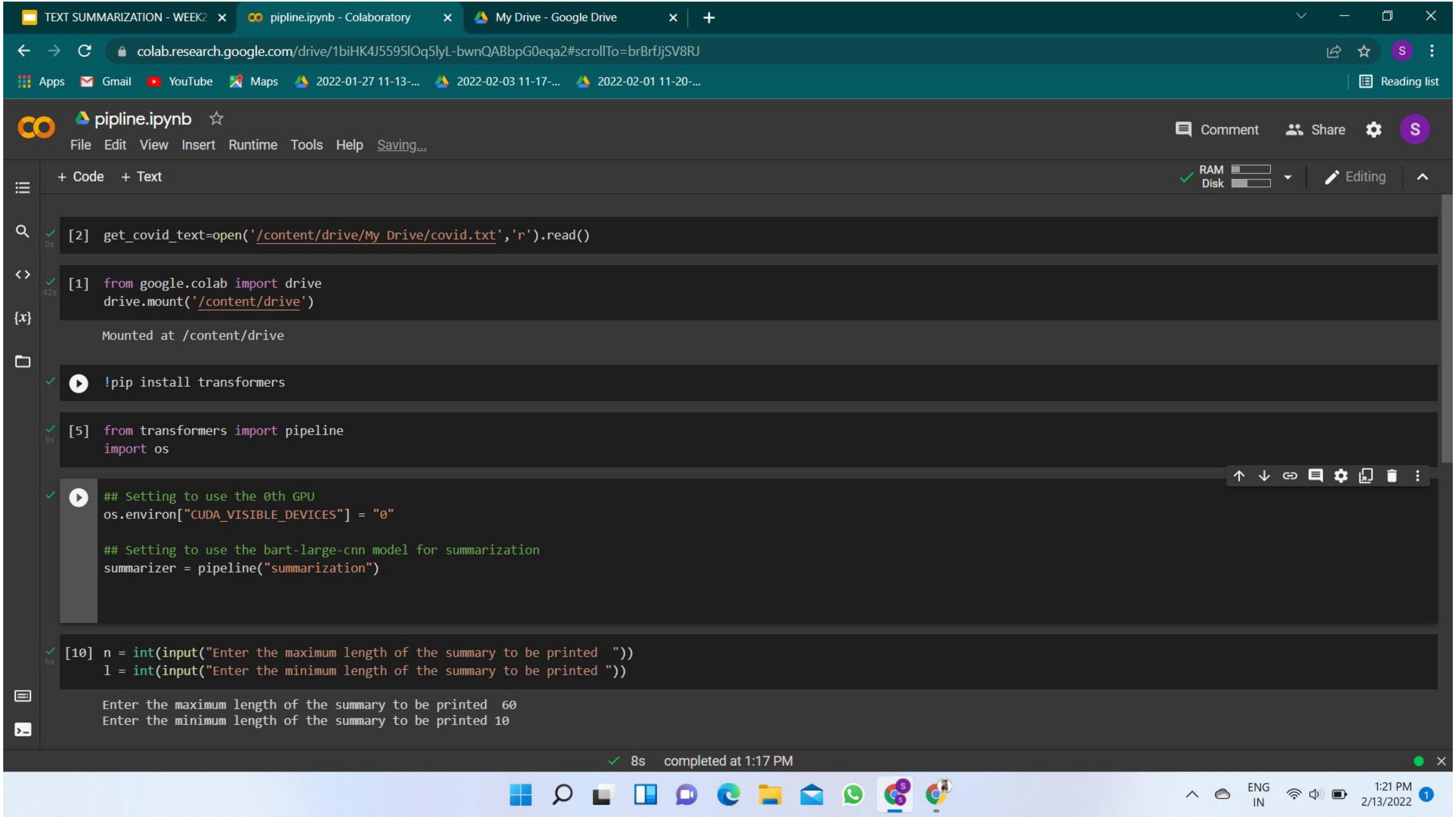
So firstly BART stands for Bidirectional and Auto Regressive Transformers. It essentially generalizes BERT and GPT based architectures by using the standard Seq2Seq. It uses a standard seq2seq model architecture combining an encoder similar to BERT and a GPT-like decoder. The pre-training task involves changing the order of the original phrases randomly and a new scheme where text ranges are switched with a single mask token. The large model of BART consists of twice as many layers as are present in the base model. It is quite similar to the BERT model but BART contains about 10% more features than the BERT model of comparable size.

BART's decoder is autoregressive, and it is regulated for generating sequential NLP tasks such as text summarization. The data is taken from the input but changed, which is closely related to the denoising pre-training objective. Hence, the input sequence embedding is the input of the encoder, and the decoder autoregressive produces output.Hence words are encoded differently depending on their position in the sentence.

In the schema on the next slide, we visualize what BART looks like at a high level. First of all, you can see that input texts are passed through the *bidirectional encoder*, i.e. the BERT-like encoder. By consequence, texts are looked at from left-to-right and right-to-left, and the subsequent output is used in the *autoregressive decoder*, which predicts the output based on the encoder input *and* the output tokens predicted so far. In other words, with BART, we can now both *understand* the inputs really well and generate new outputs.


Predicted output (e.g. summary)
Bidirectional Encoder
Autoregressive Decoder
Input (e.g. full text)

Abstractive summarization using pipeline


TEXT SUMMARIZATION - WEEK2
pipline.ipynb - Colaboratory
My Drive - Google Drive
colab.research.google.com/drive/1biHK4J5595lOq5lyL-bwnQABbpG0eqa2#scrollTo=brBrfJjSV8RJ
Apps
Gmail
YouTube
Maps
2022-01-27 11-13-...
2022-02-03 11-17-...
2022-02-01 11-20-...
Reading list
pipline.ipynb
File Edit View Insert Runtime Tools Help Saving...
Comment
Share
+ Code + Text
RAM
Disk
Editing

```
[2] get_covid_text=open('/content/drive/My Drive/covid.txt','r').read()
```

```
[1] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

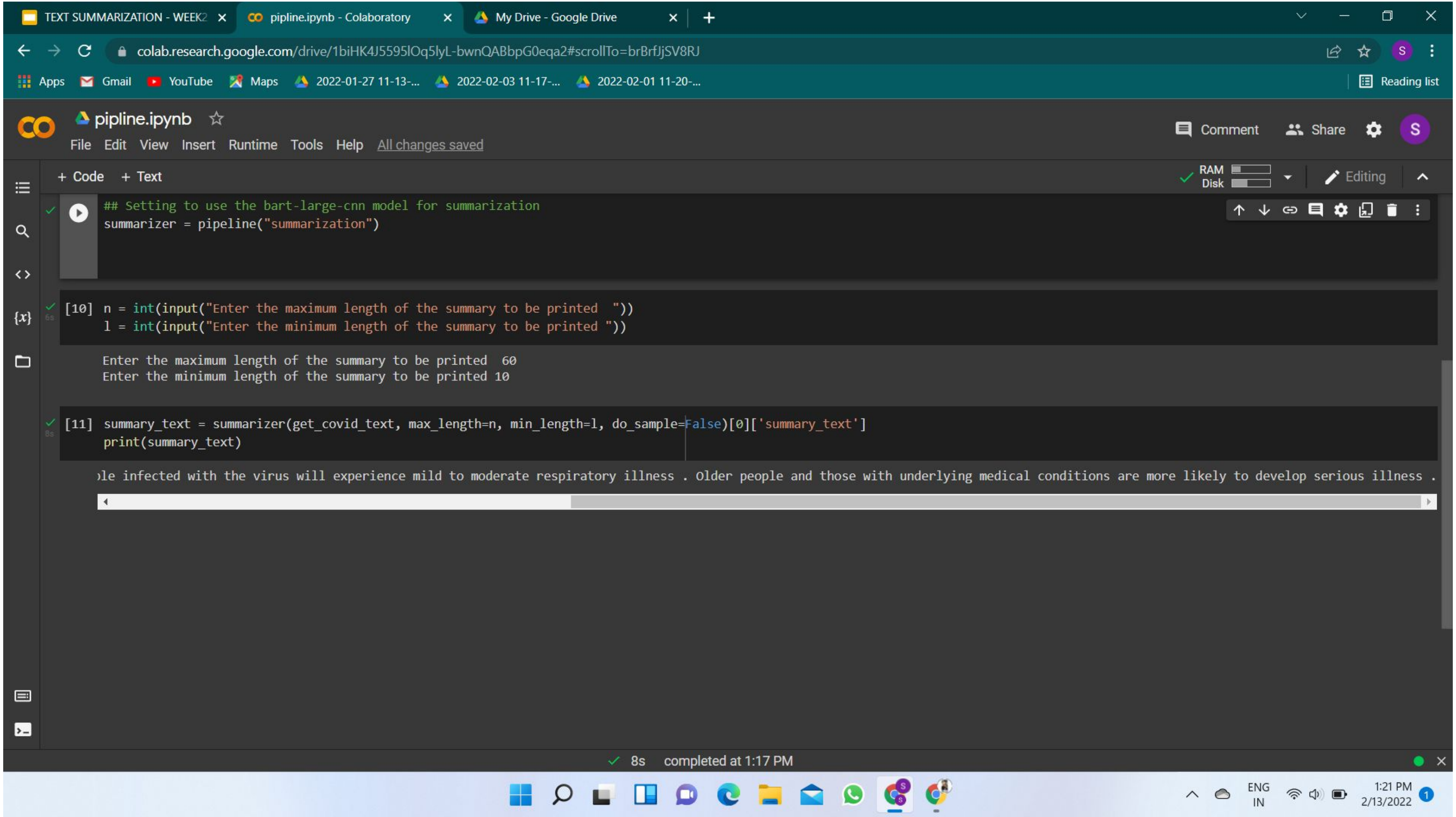
```
!pip install transformers
```

```
[5] from transformers import pipeline
    import os
```

```
## Setting to use the 0th GPU
os.environ["CUDA_VISIBLE_DEVICES"] = "0"

## Setting to use the bart-large-cnn model for summarization
summarizer = pipeline("summarization")
```

```
[10] n = int(input("Enter the maximum length of the summary to be printed  "))
     l = int(input("Enter the minimum length of the summary to be printed "))
```

```
Enter the maximum length of the summary to be printed  60
Enter the minimum length of the summary to be printed 10
```



```
## Setting to use the bart-large-cnn model for summarization
summarizer = pipeline("summarization")
```

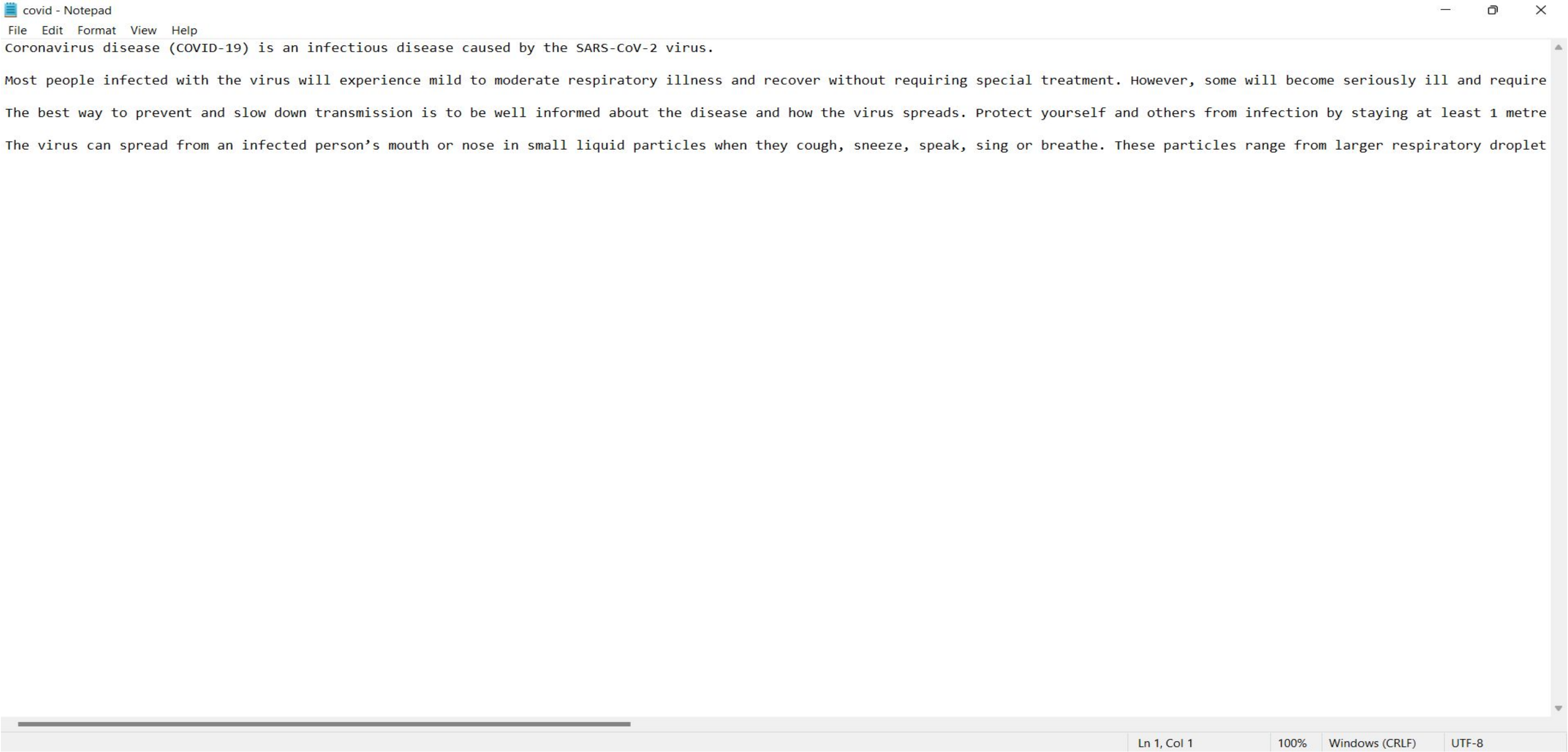
```
[10] n = int(input("Enter the maximum length of the summary to be printed  "))
     l = int(input("Enter the minimum length of the summary to be printed "))
```

```
Enter the maximum length of the summary to be printed  60
Enter the minimum length of the summary to be printed 10
```

```
[11] summary_text = summarizer(get_covid_text, max_length=n, min_length=l, do_sample=False)[0]['summary_text']
     print(summary_text)
```

```
ɔle infected with the virus will experience mild to moderate respiratory illness . Older people and those with underlying medical conditions are more likely to develop serious illness .
```

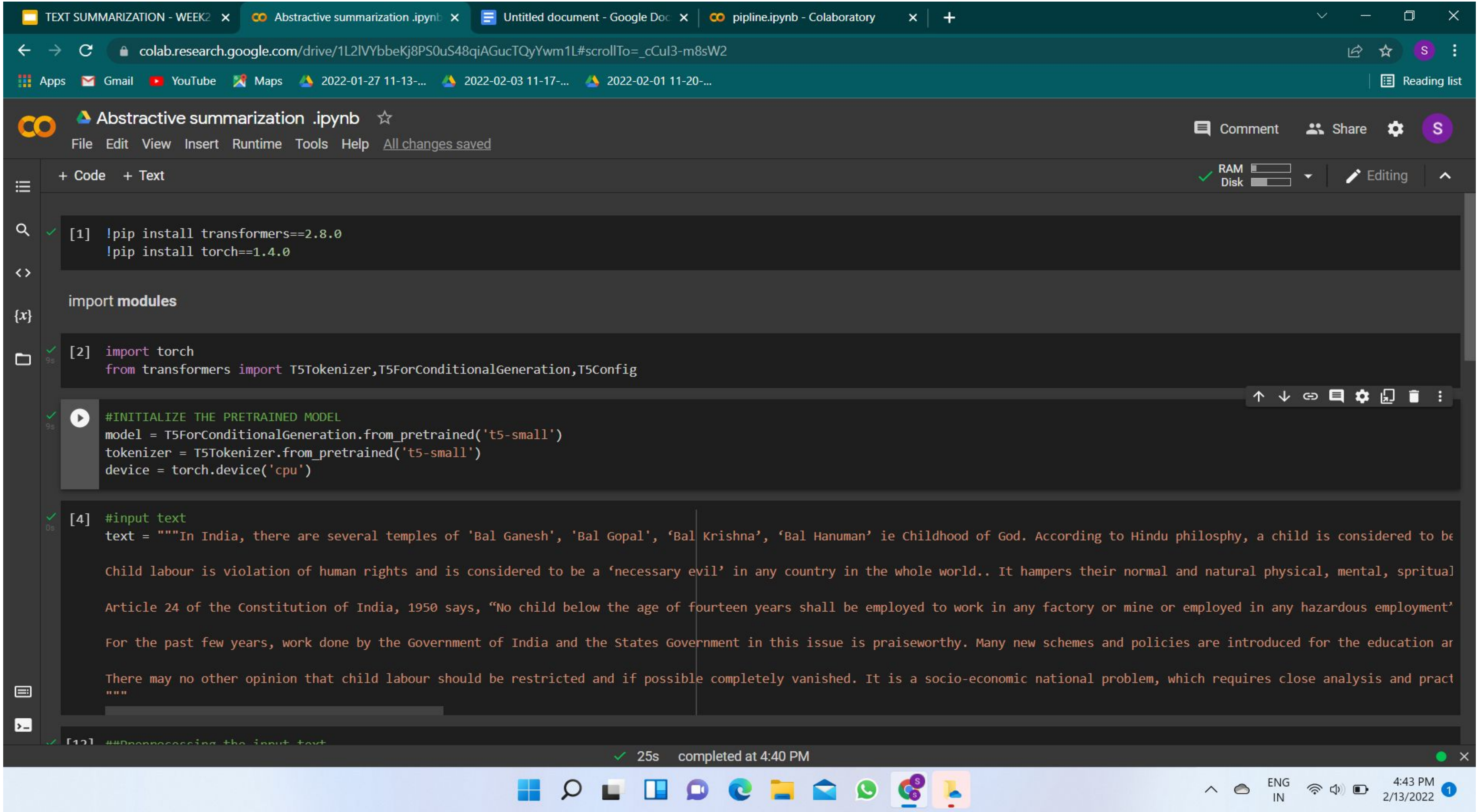
Text file



```
Coronavirus disease (COVID-19) is an infectious disease caused by the SARS-CoV-2 virus.

Most people infected with the virus will experience mild to moderate respiratory illness and recover without requiring special treatment. However, some will become seriously ill and require

The best way to prevent and slow down transmission is to be well informed about the disease and how the virus spreads. Protect yourself and others from infection by staying at least 1 metre

The virus can spread from an infected person’s mouth or nose in small liquid particles when they cough, sneeze, speak, sing or breathe. These particles range from larger respiratory droplet
```

- **T5** - T5 is the abbreviation for "Text-to-Text Transfer Transformer" . The idea behind the T5 model is transfer learning . The model was initially trained on a task containing large text in Transfer Learning before it was finely tuned on a downstream task so that the model learns general-purpose skills and information to be applied to tasks such as summarization T5 uses a sequence-to-sequence generation method that feeds the encoded input via cross-attention layers to the decoder and generates the decoder output autoregressive. We have fine-tuned a T5 model , where the encoder takes an input a series of tokens which are mapped to a sequence of embeddings. A block containing two subcomponents are present in the encoder block namely, a self attention layer and feed forward network. The decoder and encoder are similar in structure, except that there's a generalized attention mechanism after every self attention layer. This allows the model to operate only on the previous outputs. The final decoder block produces an output which is fed into another layer. This final layer is a dense layer where the activation function is softmax. The weights from the output of this layer are fed into the input embedding matrix.

Abstractive summarization using T5 model


TEXT SUMMARIZATION - WEEK2
Abstractive summarization .ipynb
Untitled document - Google Doc
pipline.ipynb - Colaboratory
colab.research.google.com/drive/1L2lVYbbeKj8PS0uS48qiAGucTQyYwm1L#scrollTo=_cCuI3-m8sW2
Apps
Gmail
YouTube
Maps
2022-01-27 11-13-...
2022-02-03 11-17-...
2022-02-01 11-20-...
Reading list
Abstractive summarization .ipynb
File
Edit
View
Insert
Runtime
Tools
Help
All changes saved
Comment
Share
+ Code
+ Text
RAM
Disk
Editing
25s
completed at 4:40 PM
ENG
IN
4:43 PM
2/13/2022

```
[1]  !pip install transformers==2.8.0
     !pip install torch==1.4.0
```

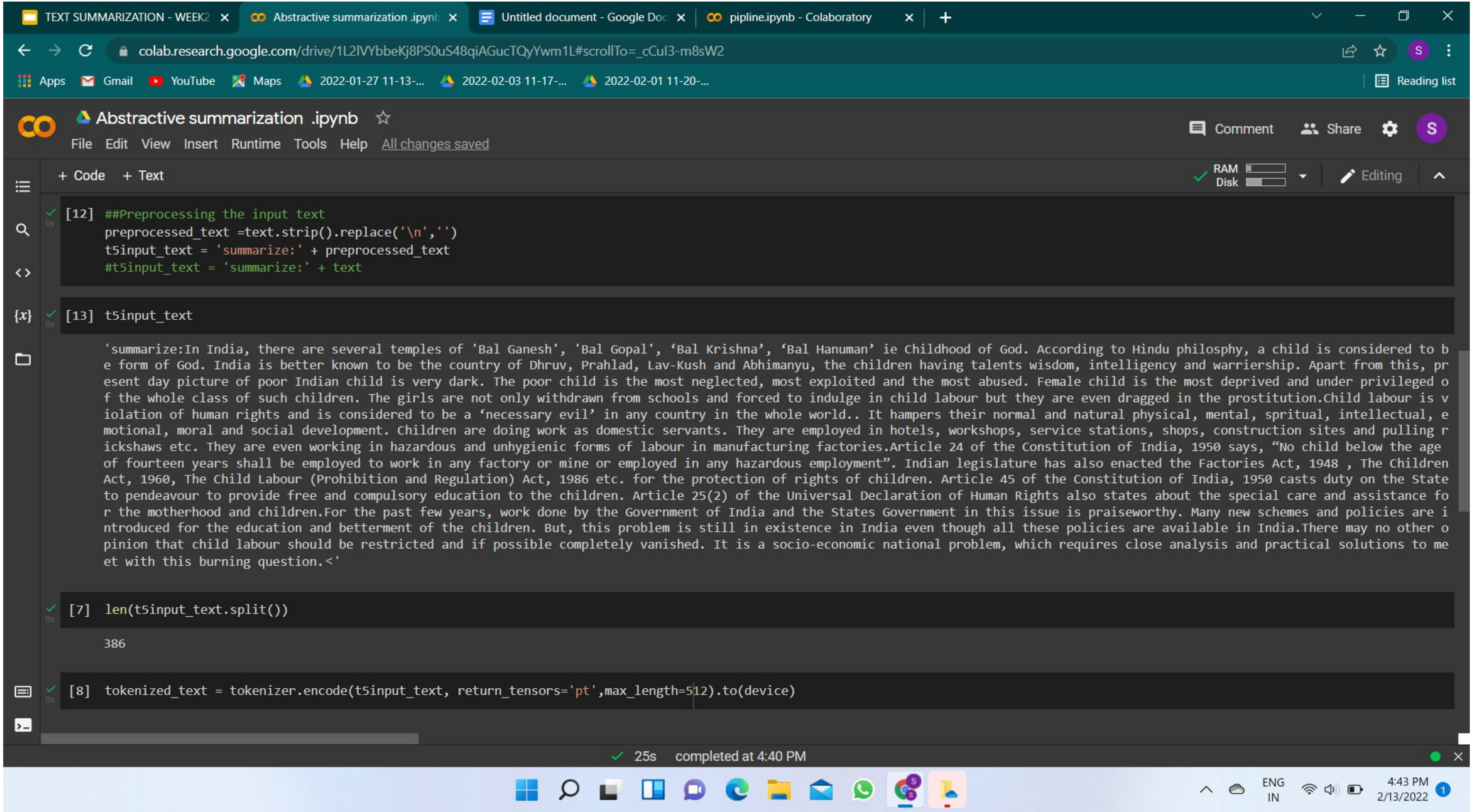
import **modules**

```
[2]  import torch
     from transformers import T5Tokenizer,T5ForConditionalGeneration,T5Config
```

```
#INITIALIZE THE PRETRAINED MODEL
model = T5ForConditionalGeneration.from_pretrained('t5-small')
tokenizer = T5Tokenizer.from_pretrained('t5-small')
device = torch.device('cpu')
```

```
[4]  #input text
     text = """In India, there are several temples of 'Bal Ganesh', 'Bal Gopal', ‘Bal Krishna’, ‘Bal Hanuman’ ie Childhood of God. According to Hindu philosphy, a child is considered to be

     Child labour is violation of human rights and is considered to be a ‘necessary evil’ in any country in the whole world.. It hampers their normal and natural physical, mental, spritual

     Article 24 of the Constitution of India, 1950 says, “No child below the age of fourteen years shall be employed to work in any factory or mine or employed in any hazardous employment”

     For the past few years, work done by the Government of India and the States Government in this issue is praiseworthy. Many new schemes and policies are introduced for the education ar

     There may no other opinion that child labour should be restricted and if possible completely vanished. It is a socio-economic national problem, which requires close analysis and pract
     """
```

```
[12] ##Preprocessing the input text
```


TEXT SUMMARIZATION - WEEK2
Abstractive summarization .ipynb
Untitled document - Google Doc
pipline.ipynb - Colaboratory
colab.research.google.com/drive/1L2lVYbbeKj8PS0uS48qiAGucTQyYwm1L#scrollTo=_cCul3-m8sW2
Apps
Gmail
YouTube
Maps
2022-01-27 11-13-...
2022-02-03 11-17-...
2022-02-01 11-20-...
Reading list
Abstractive summarization .ipynb
File
Edit
View
Insert
Runtime
Tools
Help
All changes saved
Comment
Share
+ Code
+ Text
RAM
Disk
Editing
25s
completed at 4:40 PM
ENG
IN
4:43 PM
2/13/2022

```
[12] ##Preprocessing the input text
     preprocessed_text =text.strip().replace('\n','')
     t5input_text = 'summarize:' + preprocessed_text
     #t5input_text = 'summarize:' + text
```

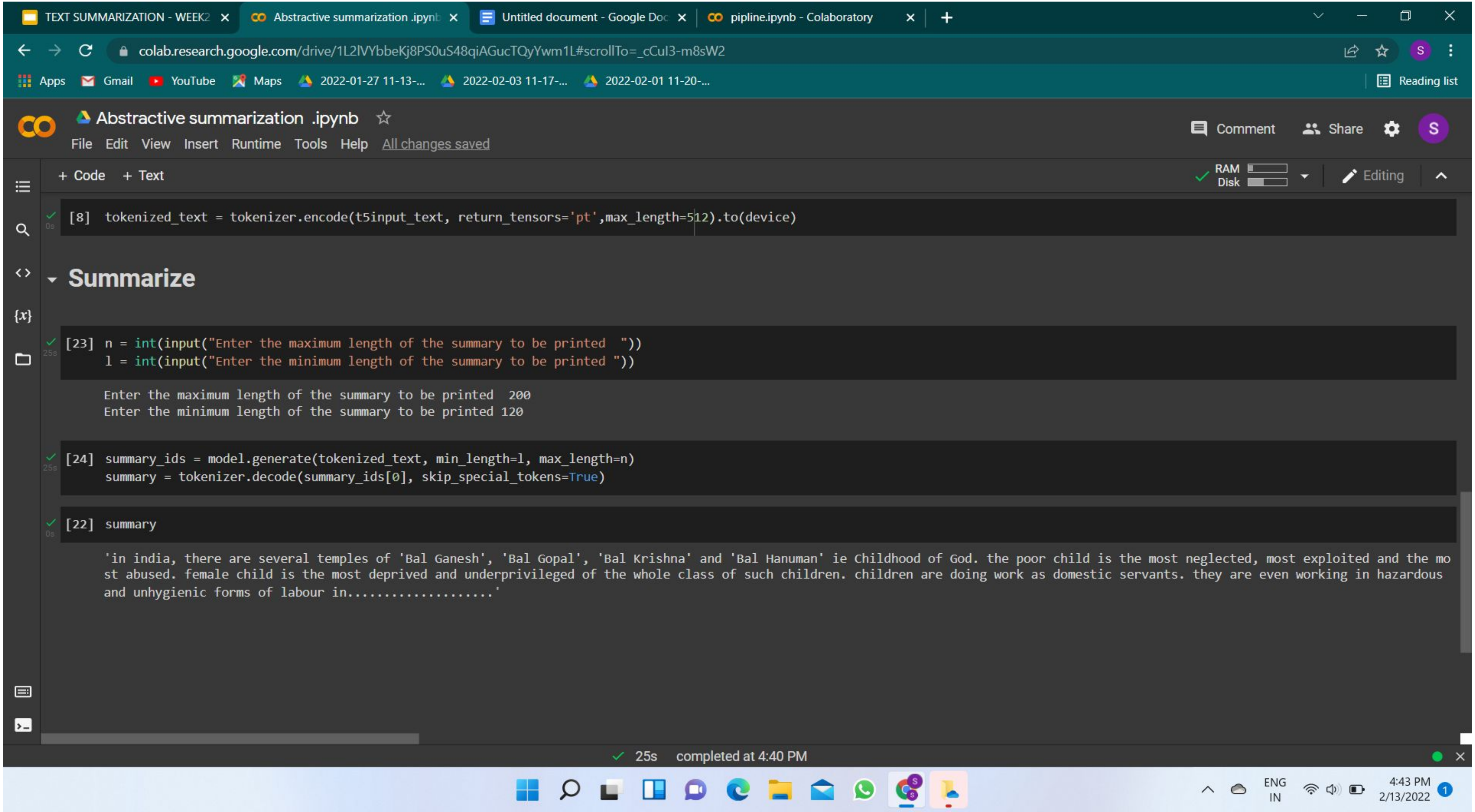
```
[13] t5input_text
```

```
'summarize:In India, there are several temples of 'Bal Ganesh', 'Bal Gopal', ‘Bal Krishna’, ‘Bal Hanuman’ ie Childhood of God. According to Hindu philosphy, a child is considered to b
e form of God. India is better known to be the country of Dhruv, Prahlad, Lav-Kush and Abhimanyu, the children having talents wisdom, intelligency and warriership. Apart from this, pr
esent day picture of poor Indian child is very dark. The poor child is the most neglected, most exploited and the most abused. Female child is the most deprived and under privileged o
f the whole class of such children. The girls are not only withdrawn from schools and forced to indulge in child labour but they are even dragged in the prostitution.Child labour is v
iolation of human rights and is considered to be a ‘necessary evil’ in any country in the whole world.. It hampers their normal and natural physical, mental, spritual, intellectual, e
motional, moral and social development. Children are doing work as domestic servants. They are employed in hotels, workshops, service stations, shops, construction sites and pulling r
ickshaws etc. They are even working in hazardous and unhygienic forms of labour in manufacturing factories.Article 24 of the Constitution of India, 1950 says, “No child below the age
of fourteen years shall be employed to work in any factory or mine or employed in any hazardous employment”. Indian legislature has also enacted the Factories Act, 1948 , The Children
Act, 1960, The Child Labour (Prohibition and Regulation) Act, 1986 etc. for the protection of rights of children. Article 45 of the Constitution of India, 1950 casts duty on the State
to pendeavour to provide free and compulsory education to the children. Article 25(2) of the Universal Declaration of Human Rights also states about the special care and assistance fo
r the motherhood and children.For the past few years, work done by the Government of India and the States Government in this issue is praiseworthy. Many new schemes and policies are i
ntroduced for the education and betterment of the children. But, this problem is still in existence in India even though all these policies are available in India.There may no other o
pinion that child labour should be restricted and if possible completely vanished. It is a socio-economic national problem, which requires close analysis and practical solutions to me
et with this burning question.<'
```

```
[7] len(t5input_text.split())
```

```
386
```

```
[8] tokenized_text = tokenizer.encode(t5input_text, return_tensors='pt',max_length=512).to(device)
```


TEXT SUMMARIZATION - WEEK2
Abstractive summarization .ipynb
Untitled document - Google Doc
pipline.ipynb - Colaboratory
colab.research.google.com/drive/1L2lVYbbeKj8PS0uS48qiAGucTQyYwm1L#scrollTo=_cCul3-m8sW2
Apps
Gmail
YouTube
Maps
2022-01-27 11-13-...
2022-02-03 11-17-...
2022-02-01 11-20-...
Reading list
Abstractive summarization .ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
25s completed at 4:40 PM
ENG
IN
4:43 PM
2/13/2022

```
[8]  tokenized_text = tokenizer.encode(t5input_text, return_tensors='pt',max_length=512).to(device)
```

Summarize

```
[23] n = int(input("Enter the maximum length of the summary to be printed  "))
     l = int(input("Enter the minimum length of the summary to be printed "))

     Enter the maximum length of the summary to be printed  200
     Enter the minimum length of the summary to be printed 120
```

```
[24] summary_ids = model.generate(tokenized_text, min_length=l, max_length=n)
     summary = tokenizer.decode(summary_ids[0], skip_special_tokens=True)
```

```
[22] summary

     'in india, there are several temples of 'Bal Ganesh', 'Bal Gopal', 'Bal Krishna' and 'Bal Hanuman' ie Childhood of God. the poor child is the most neglected, most exploited and the mo
     st abused. female child is the most deprived and underprivileged of the whole class of such children. children are doing work as domestic servants. they are even working in hazardous
     and unhygienic forms of labour in...................'
```

DIFFERENCE BETWEEN PIPELINE AND T5 MODEL

- The summaries generated by the pipeline model included sentences that deviated the most compared to the original reference summary and focused on unimportant sentences.
- The T5 model shows good results and a comparatively higher Rogue score F value. The summaries generated are coherent and accurate. The text meaning was preserved in these summaries and aligned well with the original summary.

Some Real life applications

- Applications such as search engines and news websites use text summarisation. In search engines, previews are produced as snippets, and news websites generate headlines to describe the news to facilitate knowledge retrieval.
- **Internal document workflow -** Large companies are constantly producing internal knowledge, which frequently gets stored and under-used in databases as unstructured data. Summarization can enable analysts to quickly understand everything the company has already done in a given subject, and quickly assemble reports that incorporate different points of view.
- **Social media marketing -**Companies producing long-form content, like whitepapers, e-books and blogs, might be able to leverage summarization to break down this content and make it shareable on social media sites like Twitter or Facebook. This would allow companies to further re-use existing content.
- **Books and literature -**Google has reportedly worked on projects that attempt to understand novels. Summarization can help consumers quickly understand what a book is about as part of their buying process.

THANK YOU!!