”Sarcasm Detection using Dilbert and Albert: An In-Depth Comparative Analysis with Bert”

MSc Research Project
Programme Name

Rohit Gopal Wadhwani
Student ID: 21194645

School of Computing
National College of Ireland

Supervisor: Dr.Muslim Jameel Syed

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Rohit Gopal Wadhwani
Student ID:	21194645
Programme:	Programme Name
Year:	2023
Module:	MSc Research Project
Supervisor:	Dr.Muslim Jameel Syed
Submission Due Date:	14/08/2023
Project Title:	”Sarcasm Detection using Dilbert and Albert: An In-Depth Comparative Analysis with Bert”
Word Count:	931
Page Count:	14

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Rohit Gopal Wadhwani
Date:	16th September 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

”Sarcasm Detection using Dilbert and Albert: An In-Depth Comparative Analysis with Bert”

Rohit Gopal Wadhwani
21194645

1 Introduction

This configuration manual is a thorough reference to the setup, implementation, and deployment of the sarcasm detection project. It describes the system requirements, installation stages, data preprocessing approaches, model architecture considerations, training and evaluation procedures, and guidelines for different deployment situations.

By following the directions in this manual, you will get insight into the fundamental principles of creating, training, and analyzing deep learning models for sarcasm detection.

2 Software Description

Category	**Tools/Services**
Programming	Python Python Libraries Google Colaboratory
Report	Overleaf (LATEX format)
Web Browsers	Google Chrome

Table 1: Tools and Services

3 Setting Up Environment

I chose to create our deep learning models using ”Google Colab,” a computing platform. Google Colab is a web-based application created by Google *Google Colab* (n.d.). It’s especially well-suited for working with sophisticated models, such as deep learning models, which frequently need a large amount of computer power. A strong machine would be required if I used a local configuration, such as a Jupyter notebook. However, by using Google Colab, we can gain free access to cloud-based GPUs, which substantially speeds up our model implementation process.In our case we have also used T4 GPU Runtime as this helps our models run 10x times and saved a lot of our time. But getting T4 GPU is very subjective as we are using free version of Google Colab. Here is the steps and figures 1 for detail information.

Step1 : Go to https://colab.google/

Step2: Click on New Notebook

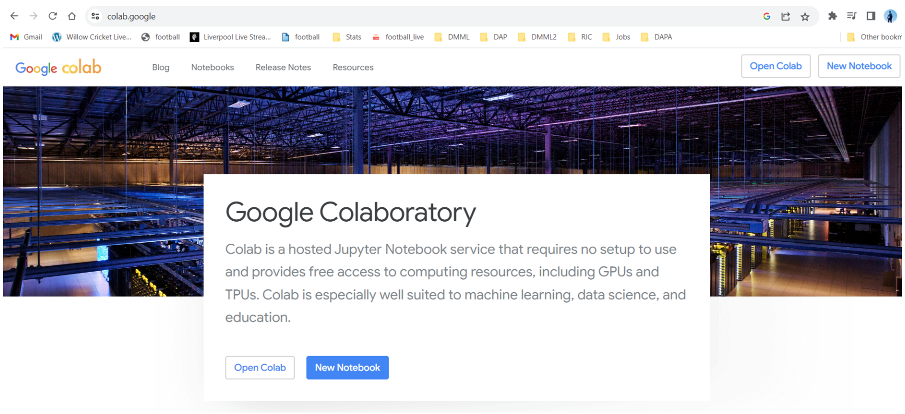


Figure 1: Google Colab

Step3 : Please change Runtime Type to Python3 and Hardware accelerator T4 GPU 2 3 which will make your deep learning models run more faster than normal CPU.

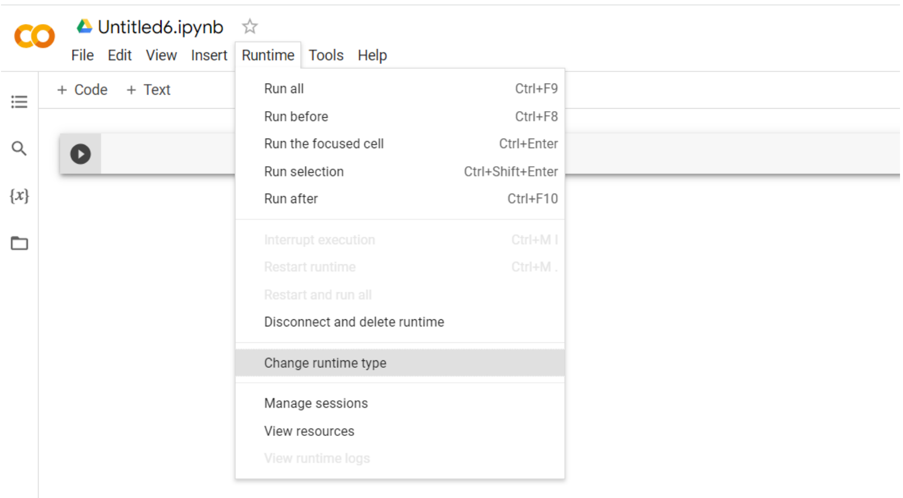


Figure 2: Runtine Settings

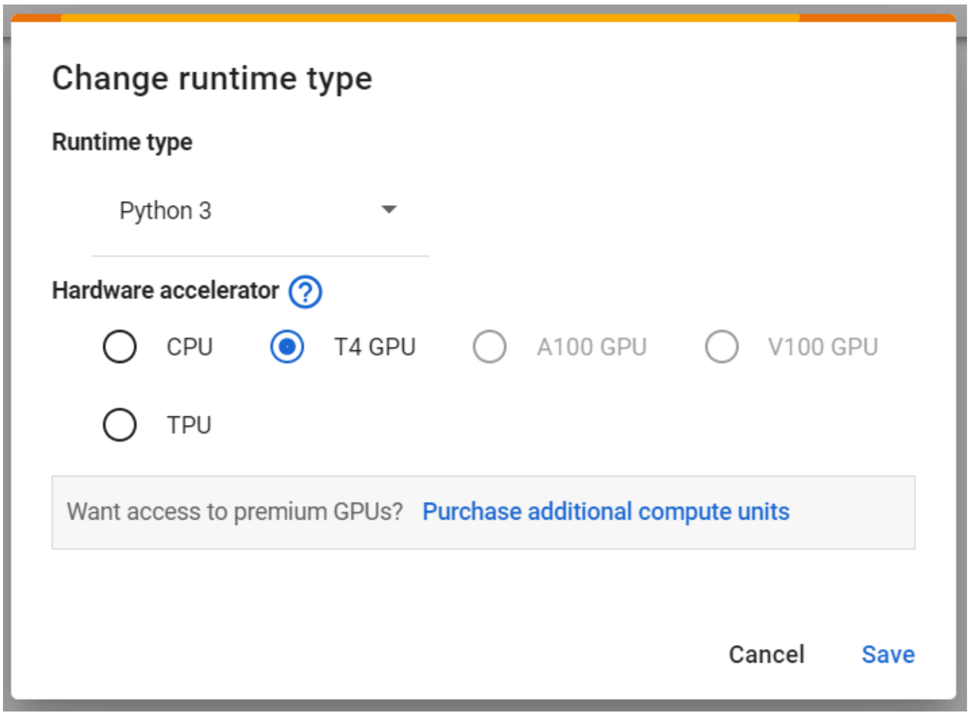


Figure 3: Runtine Settings 2

4 Dataset of News Headlines

4.1 Data Gathering

1. To import the dataset into the Python environment of Google Colab, first obtain the dataset of "News Headlines Dataset For Sarcasm Detection" (Mishra, 2019) from the website "https://www.kaggle.com" *News Headlines Dataset for Sarcasm Detection* (n.d.)(Mishra, News Headlines Dataset For Sarcasm Detection, 2019).

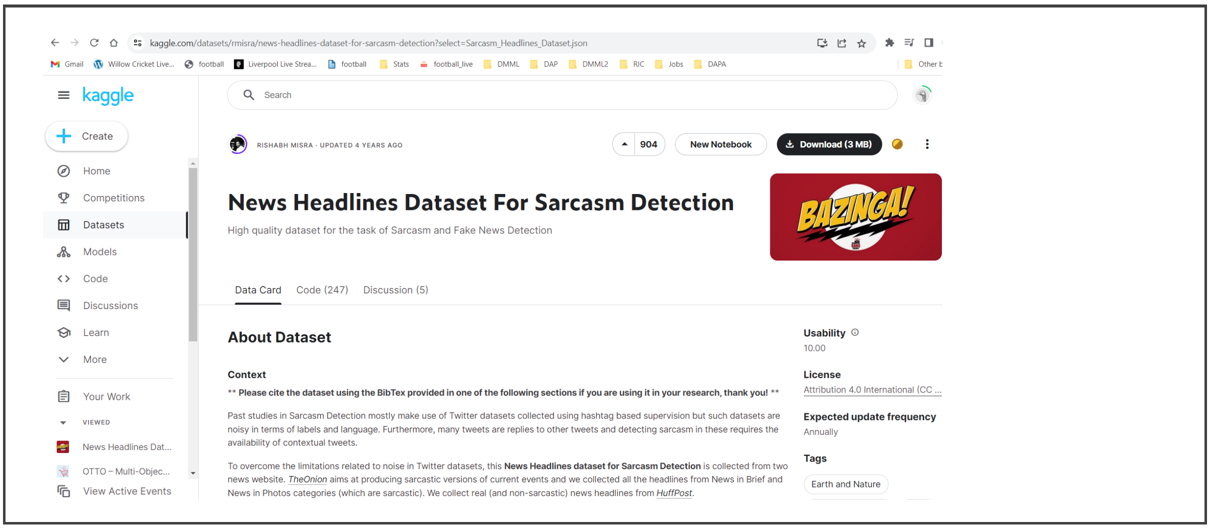


Figure 4: Dataset

2. After downloading, save the dataset to Google Drive for future use in Google Colab, or save it locally and upload it to Google Colab when starting a new notebook.

3. I had personally saved the dataset in my local computer and then uploaded it as shown in the figure 5

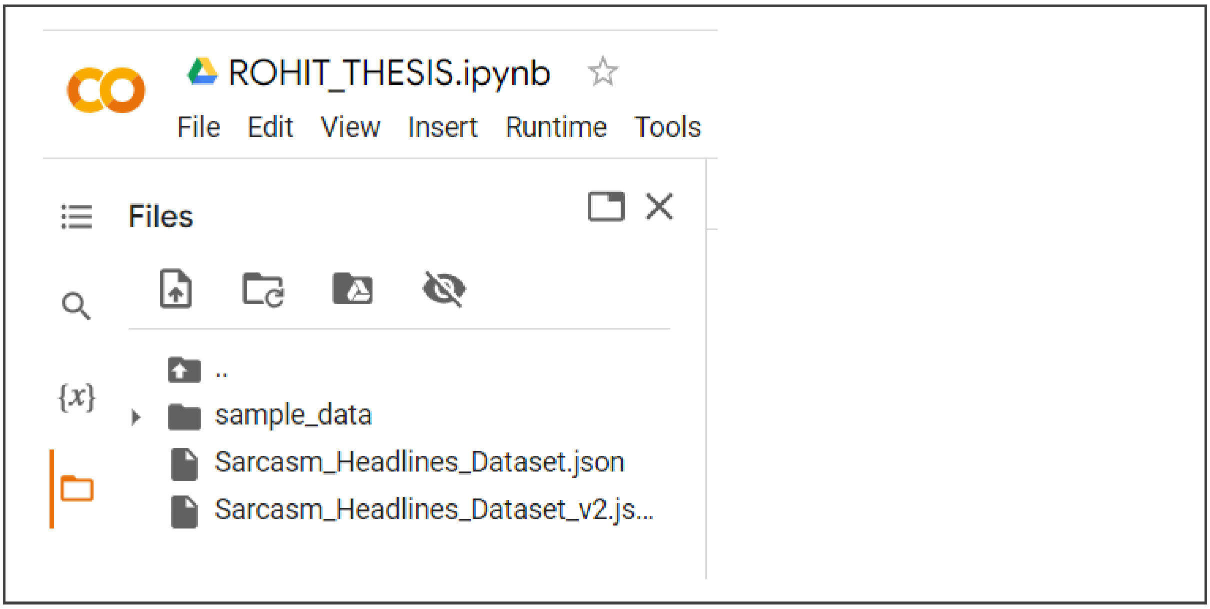


Figure 5: File Upload

4.2 Importing Python Libraries

1.There are many libraries in python which we can import it and it helps us to code efficiently , libraries used in this research is shown in the figure 6.

2. After Importing such libraries according to our need , we can move forward with using such libraries in our research.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px

import nltk
from sklearn.feature_extraction.text import CountVectorizer
from nltk.corpus import stopwords
from wordcloud import WordCloud,STOPWORDS
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
from bs4 import BeautifulSoup
import re,string,unicodedata
from sklearn.metrics import classification_report,confusion_matrix,accuracy_score,f1_score
from sklearn.model_selection import train_test_split
from string import punctuation
from nltk import pos_tag
from nltk.corpus import wordnet


import keras
import tensorflow as tf
import tensorflow_hub as hub
from tensorflow import keras
from keras import backend as K
from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.optimizers import Adam
from keras.layers import LSTM,Dense,Bidirectional,Input
from keras.models import Model
import torch
!pip install transformers
!pip install imbalanced-learn

!pip install sentencepiece
```

Figure 6: Libraries

4.3 Data Preprocessing and Analyzing

Step 1 : We will First look into the data and will convert our json file to Data frames and merge both the json files into 1

Step 2 : we will only keep the important variables and remove the rest as part of data cleaning

Figure 7: Data Pre-processing

Step 3 : As a part of Data cleaning , we will then check the data frame and move forward with the shape of the data set which will give an idea of the rows and columns of our dataset and info will help us to check if there is any null values in our dataset as well as data type of our variables as shown in the figure 8.

```
[5] #selecting only the necessary variables, as article link is not that important
    headlines_v1 = headlines_v1[['headline','is_sarcastic']]
    headlines_v2 = headlines_v2[['headline','is_sarcastic']]

    #concating together into 1
    headlines = pd.concat([headlines_v1,headlines_v2])
    headlines.reset_index(drop=True, inplace=True)
```

Double-click (or enter) to edit

```
[6] headlines.head()
```

	headline	is_sarcastic
0	former versace store clerk sues over secret 'b...	0
1	the 'roseanne' revival catches up to our thorn...	0
2	mom starting to fear son's web series closest ...	1
3	boehner just wants wife to listen, not come up...	1
4	j.k. rowling wishes snape happy birthday in th...	0

```
[7] headlines.shape

    (55328, 2)
```

```
[8] headlines.info()

    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 55328 entries, 0 to 55327
    Data columns (total 2 columns):
     #   Column        Non-Null Count  Dtype
    ---  ------        --------------  -----
     0   headline      55328 non-null  object
     1   is_sarcastic  55328 non-null  int64
    dtypes: int64(1), object(1)
```

Figure 8: Data Cleaned

Step 4 : We have also removed the necessary Stop words removal, lemmatization, special characters removal, URL's, Square bracket, and noisy text as shown in the figure 9.

```
# Data cleaning - Stop words removal, lemmatization, special characters removal
stop_words = stopwords.words('english')
stop_words.remove('not')

def clean_text(sentence):

    headline = re.sub('\s+\n+', ' ', sentence)
    headline = re.sub('[^a-zA-Z0-9]', ' ', sentence)
    headline = headline.lower()
    headline = headline.split()

    lemm = WordNetLemmatizer()
    headline = [lemm.lemmatize(word, "v") for word in headline if not word in stop_words]
    headline = ' '.join(headline)

    return headline
```

```
[14] def strip_html(text):
         soup = BeautifulSoup(text, "html.parser")
         return soup.get_text()

     #Removing the square brackets
     def remove_between_square_brackets(text):
         return re.sub('\[[^]]*\]', '', text)
     # Removing URL's
     def remove_between_square_brackets(text):
         return re.sub(r'http\S+', '', text)
     #Removing the noisy text
     def denoise_text(text):
         text = strip_html(text)
         text = remove_between_square_brackets(text)
         text = remove_stopwords(text)
         return text
```

Figure 9: Data Cleaning

Step 5 : Word Clouds to understand which words was used the most in both the headlines that is in sarcastic headlines and in non-sarcastic headlines.

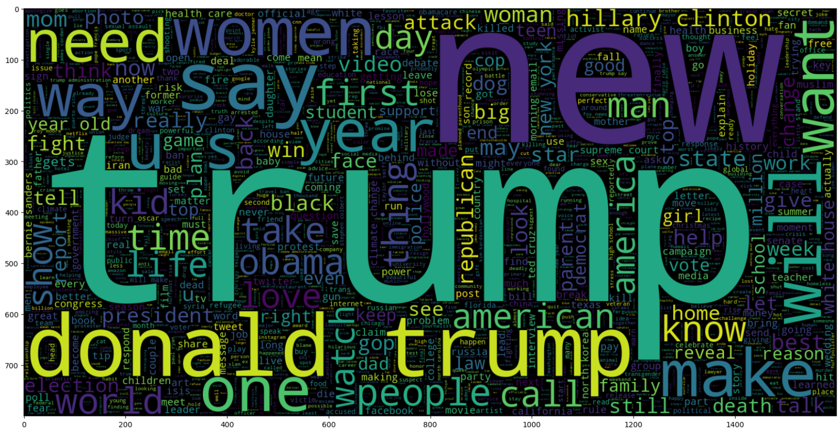


Figure 10: Word-Cloud for non- sarcastic Headlines

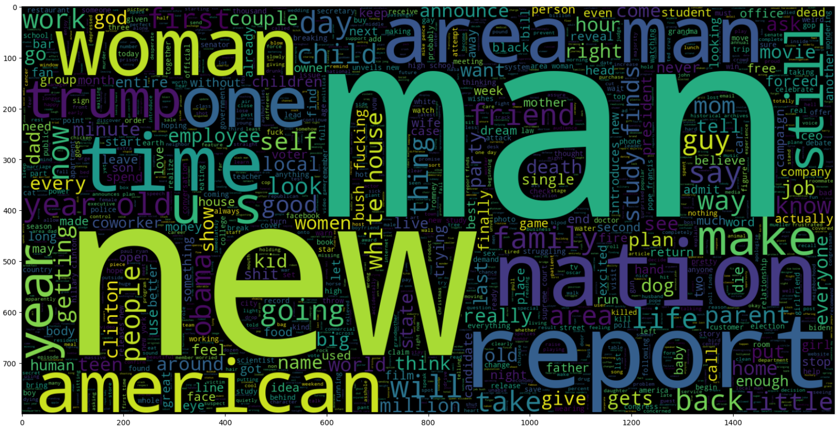


Figure 11: Word-Cloud for Sarcastic Headlines

Step 6 : To check if there is any outliers as outliers can create biasness and if there are any we will remove it to balance the data and do not create any bias towards anything.

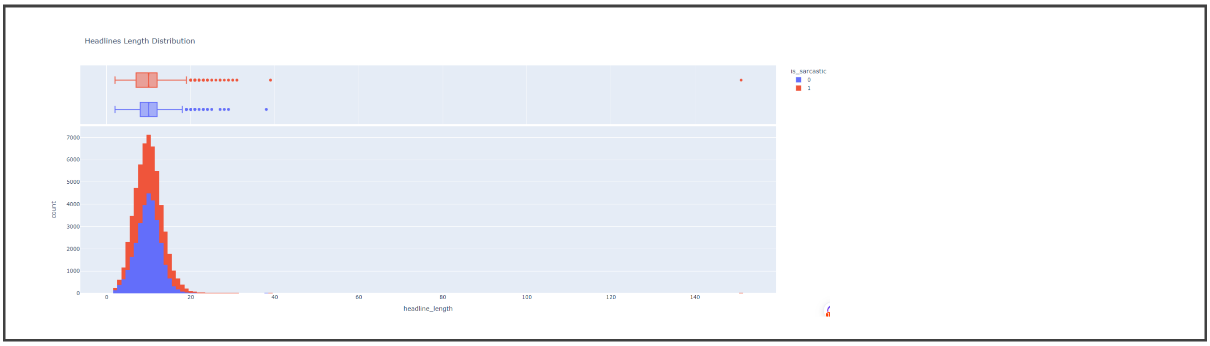


Figure 12: Outlier Check

Step 7 ; In the figure 12 we can see that we have one outlier for one sarcastic headline 13, so will remove it and then will run our models on such datasets

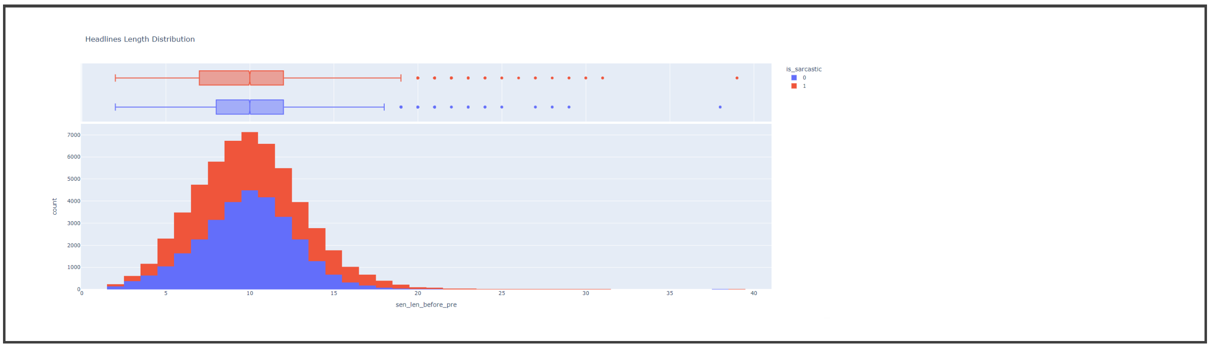


Figure 13: Outlier removed

5 Model

1. In this research we have used 3 models i.e BERT, DILBERT and ALBERT for sarcasm detection in news headlines. For this manual configuration file I will only show 1 model that is BERT example and the results to get a clear view of my research. I have also used the same architecture and the evalution metrics for the other models. For pre processing I have taken help from Abadi et al. (2016), and Gosavi (2022).

```
[27] import tensorflow as tf
     import matplotlib.pyplot as plt
     from transformers import TFBertModel, BertTokenizer
     from sklearn.model_selection import train_test_split

     labels = data.is_sarcastic.values
     sentences = data.headline.values

     PRE_TRAINED_MODEL_NAME = 'bert-base-uncased'
     tokenizer = BertTokenizer.from_pretrained(PRE_TRAINED_MODEL_NAME,do_lower_case = True)

     def encoder(sentences):
       ids = []
       for sentence in sentences:
         encoding = tokenizer.encode_plus(
         sentence,
         max_length=16,
         truncation = True,
         add_special_tokens=True,
         return_token_type_ids=False,
         pad_to_max_length=True,
         return_attention_mask=False)
         ids.append(encoding['input_ids'])
       return ids

     #Train test split
     train_sents,test_sents, train_labels, test_labels  = train_test_split(sentences,labels,test_size=0.15)

     train_ids = encoder(train_sents)
     test_ids = encoder(test_sents)

Downloading (…)solve/main/vocab.txt: 100%  232k/232k [00:00<00:00, 9.37MB/s]
Downloading (…)okenizer_config.json: 100%  28.0/28.0 [00:00<00:00, 1.05kB/s]
Downloading (…)lve/main/config.json: 100%  570/570 [00:00<00:00, 23.0kB/s]
```

Figure 14: BERT Model

The figure 14 shows how to use BERT's tokenizer to preprocess text data. It encodes headlines into numerical token sequences to prepare the data for a binary classification task (sarcastic or not). The dataset is then divided into two parts: training and testing. The encoder function encodes each headline into fixed-length sequences suitable for model input using BERT's tokenizer. The encoded sequences and labels are separated into training and test sets, creating down the foundation for creating, training, and testing a classification model.

```
[28] # The patience parameter is the amount of epochs to check for improvement
     early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=5)


     train_ids = tf.convert_to_tensor(train_ids)
     test_ids = tf.convert_to_tensor(test_ids)
     test_labels = tf.convert_to_tensor(test_labels)
     train_labels = tf.convert_to_tensor(train_labels)
```

```
bert_encoder = TFBertModel.from_pretrained('bert-base-uncased')
input_word_ids = tf.keras.Input(shape=(16,), dtype=tf.int32, name="input_word_ids")
embedding = bert_encoder([input_word_ids])
dense = tf.keras.layers.Lambda(lambda seq: seq[:, 0, :])(embedding[0])
dense = tf.keras.layers.Dense(128, activation='relu')(dense)
dense = tf.keras.layers.Dropout(0.2)(dense)
output = tf.keras.layers.Dense(1, activation='sigmoid')(dense)

model = tf.keras.Model(inputs=[input_word_ids], outputs=output)
```

```
Downloading model.safetensors: 100% 440M/440M [00:01<00:00, 357MB/s]
```

Figure 15: Early Stopping- BERT Model

```
model.compile(tf.keras.optimizers.Adam(1e-5), loss='binary_crossentropy', metrics=['accuracy'])
model.summary()
```

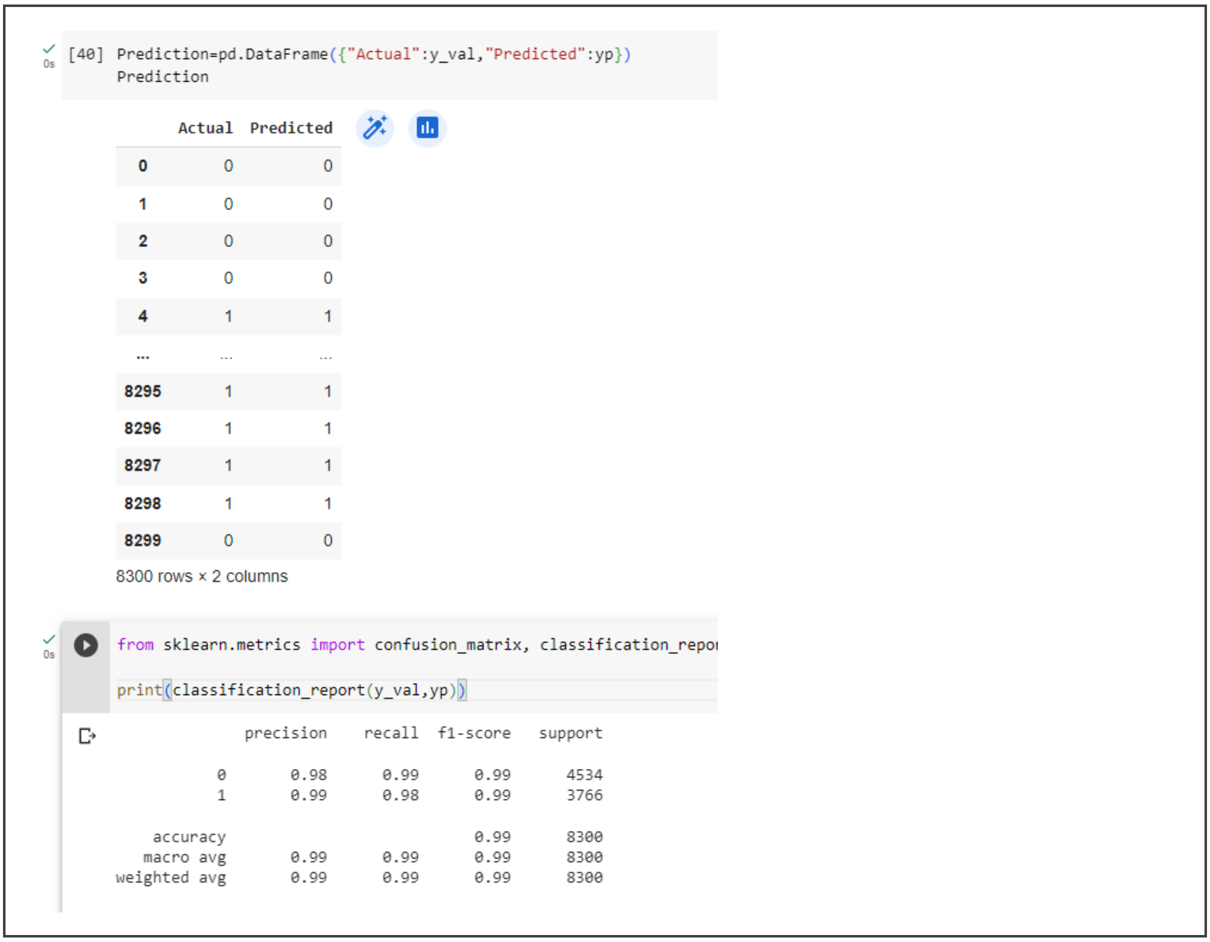
```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_word_ids (InputLayer)  [(None, 16)]             0

 tf_bert_model (TFBertModel)  TFBaseModelOutputWithPoo  109482240
                             lingAndCrossAttentions(l
                             ast_hidden_state=(None,
                             16, 768),
                              pooler_output=(None, 76
                             8),
                              past_key_values=None, h
                             idden_states=None, atten
                             tions=None, cross_attent
                             ions=None)

 lambda (Lambda)             (None, 768)               0

 dense (Dense)               (None, 128)               98432

 dropout_37 (Dropout)        (None, 128)               0

 dense_1 (Dense)             (None, 1)                 129

=================================================================
Total params: 109,580,801
Trainable params: 109,580,801
Non-trainable params: 0
_________________________________________________________________
```

```
[31] history = model.fit(x = train_ids, y = train_labels, epochs = 20, verbose = 1, batch_size = 64, callbacks=[early_stop], validation_data = (test_ids, test_labels))
```

Figure 16: BERT Model- Compile

5.1 Results

We have taken Results and evaluated it using Confusion matrix and ROC curves



```
[40] Prediction=pd.DataFrame({"Actual":y_val,"Predicted":yp})
     Prediction
```

	Actual	Predicted
0	0	0
1	0	0
2	0	0
3	0	0
4	1	1
...	...	...
8295	1	1
8296	1	1
8297	1	1
8298	1	1
8299	0	0

8300 rows × 2 columns

```
from sklearn.metrics import confusion_matrix, classification_repor

print(classification_report(y_val,yp))
```

```
              precision    recall  f1-score   support

           0       0.98      0.99      0.99      4534
           1       0.99      0.98      0.99      3766

    accuracy                           0.99      8300
   macro avg       0.99      0.99      0.99      8300
weighted avg       0.99      0.99      0.99      8300
```

Figure 17: Results for BERT Model

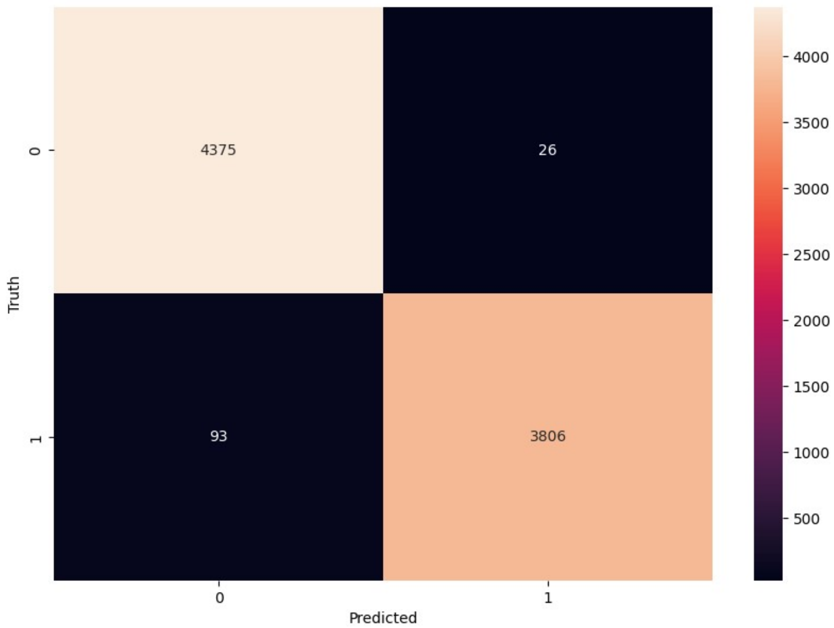


Figure 18: Confusion Matrix for BERT Model

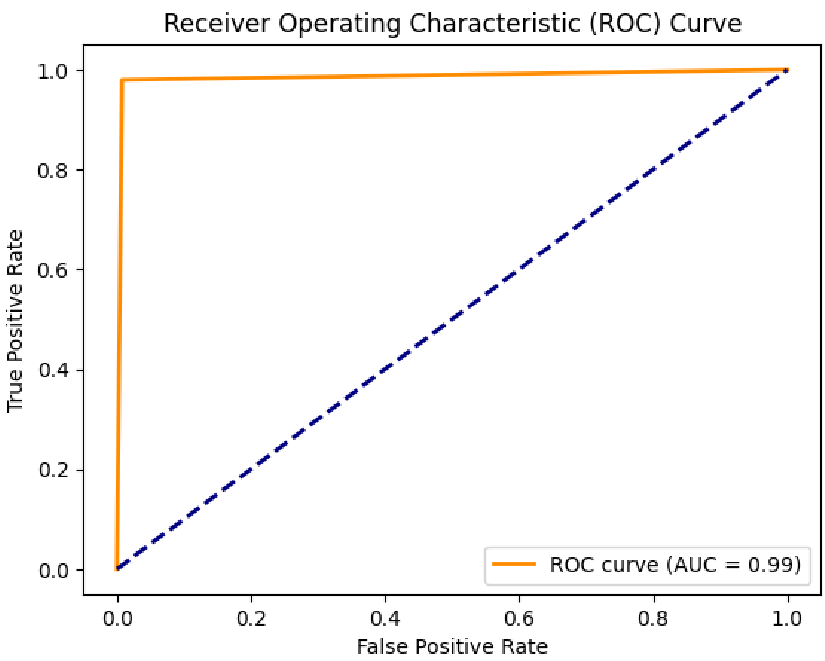


Figure 19: ROC-AUC for BERT Model

6 Video Presentation link

Click here to access the link.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M. et al. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems, *arXiv preprint arXiv:1603.04467* .

Google Colab (n.d.). `https://colab.research.google.com/`.

Gosavi, S. R. (2022). *Transformer based Detection of Sarcasm and it's Sentiment in Textual Data*, PhD thesis, Dublin, National College of Ireland.

News Headlines Dataset for Sarcasm Detection (n.d.). Kaggle Datasets. Retrieved from `https://www.kaggle.com/datasets/rmisra/news-headlines-dataset-for-sarcasm-detection?select=Sarcasm_Headlines_Dataset.json`.