Configuration Manual

# The Sentiment analysis of Movie reviews using the Transfer Learning approach

MSc Research Project
Data Analytics

Tanmay Shrikant Paranjape
Student ID: x20101015

School of Computing
National College of Ireland

Supervisor: Hicham Rifai

**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

**Student Name:** Tanmay Shrikant Paranjape

**Student ID:** X20101015

**Program:** Data Analytics **Year:** 2021-22

**Module:** Research Project

**Supervisor:** Hicham Rifai

**Submission Due Date:** 15/08/2022

**Project Title:** The Sentiment Analysis of Movie reviews using the Transfer Learning approach

**Word Count:** **1370** **Page Count:** 6

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:** Tanmay Shrikant Paranjape

**Date:** 15/08/2022

**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): | |

# Configuration Manual

Tanmay Paranjape
x20101015

## 1 Introduction

When any research project is developed, the researcher has the hands-on necessary steps followed, but it is important that the end user must get information about all these steps when the research work will get reproduced. This manual outlines the steps and practices that must be followed by every researcher before the scripts created for the current research are performed. This will allow the researchers to run the code without experiencing any difficulty and errors. Additionally, it contains all the specifications of the hardware and software used to carry out the research. This manual will be a guide to the new researchers and will help them to reproduce the research work.

## 2 System Specification

In this section, hardware and software that had been used to perform the research will be discussed. Along with that necessary packages and libraries used will be showcased.

### 2.1 Hardware requirements

Initially, the research work has been carried out using a Jupyter notebook but the time taken for execution was extremely huge. Also, many times the Jupyter notebook got crashed. Hence, to resolve this Google colab Pro has been used which provided a more robust hardware platform. The hardware used before and after has been discussed in Table 1.

**Table 1. Hardware Requirements**

| Parameters | Jupyter Notebook | Google colab pro |
|---|---|---|
| RAM | 12 GB | 32 GB |
| GPU | NVIDIA | K80 |
| Storage | 512 GB | 1 TB |
| Processor | Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz 1.80 GHz | - |
| Operating System | Windows | |

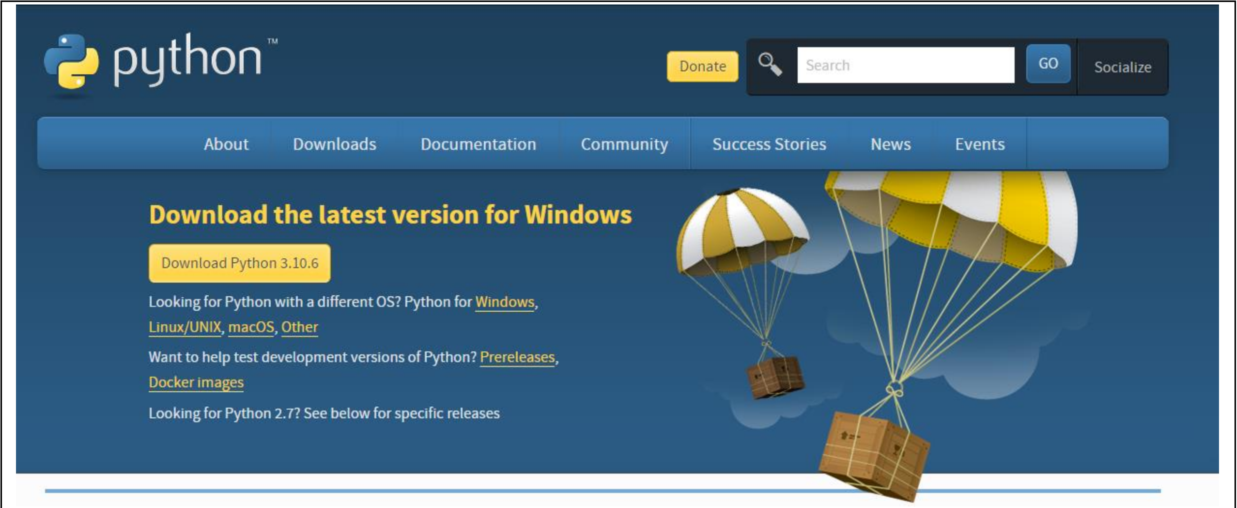
The dataset used in this research has 50,000 records and hence it became difficult to carry out the work in the Jupyter notebook. The advantage of Google colab is it loads data from Google Drive and hence it can able to fetch data within seconds. It is always recommended that the hardware specifications must be higher to lower the execution time.

## 2.2 Software requirements

The entire research has been performed using Google colab pro which is a web-based integrated development environment (IDE). Python has been used as a programming language. All the necessary steps carried out during the installation have been elaborated in the following section.

# 3 Installation and Downloads

- **Python**



Download the latest version for Windows

Looking for Python with a different OS? Python for Windows, Linux/UNIX, macOS, Other

Want to help test development versions of Python? Prereleases, Docker images

Looking for Python 2.7? See below for specific releases

**Figure 1. Web page for Python download**

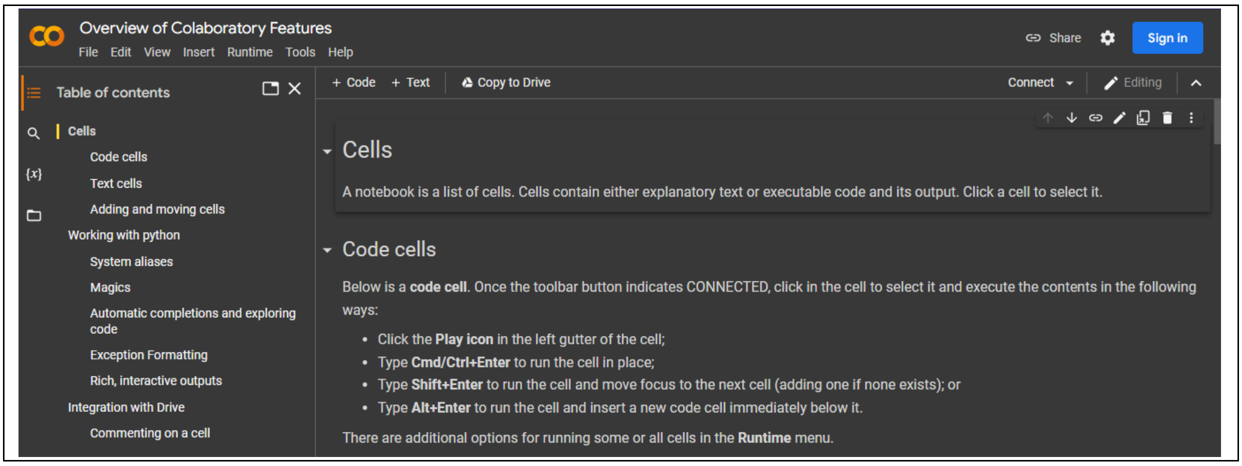
For this research project, Python has been chosen as a programming language, since by default it comes up with numerous packages and libraries. It supports the Keras and TensorFlow repositories and hence the deep learning algorithms can be used and various models can be implemented. Figure 1. represents the Python web page, by referring to which python repository[1] can be accessed. The installation guide has already been provided on the website and it supports windows, mac as well as Linux OS.

- **Google Colab**

The Google colab Pro has been used as a Web Integrated Development Environment (IDE). Initially, Jupyter notebook has been used for implementation but it failed to handle this much amount of data. The Google colab Pro[2] is having wide advantages such as, it provides additional RAM and GPU which is very important for data processing. Also all the colab files get save on Google drive hence it already has cloud storage. All the required instructions will be provided on Google colab login page and user can easily able to navigate and access the content. The Figure 2 represents Google Colaboratory login window.

[1] https://www.python.org/downloads/

[2] https://colab.research.google.com/notebooks/basic_features_overview.ipynb



Cells

A notebook is a list of cells. Cells contain either explanatory text or executable code and its output. Click a cell to select it.

Code cells

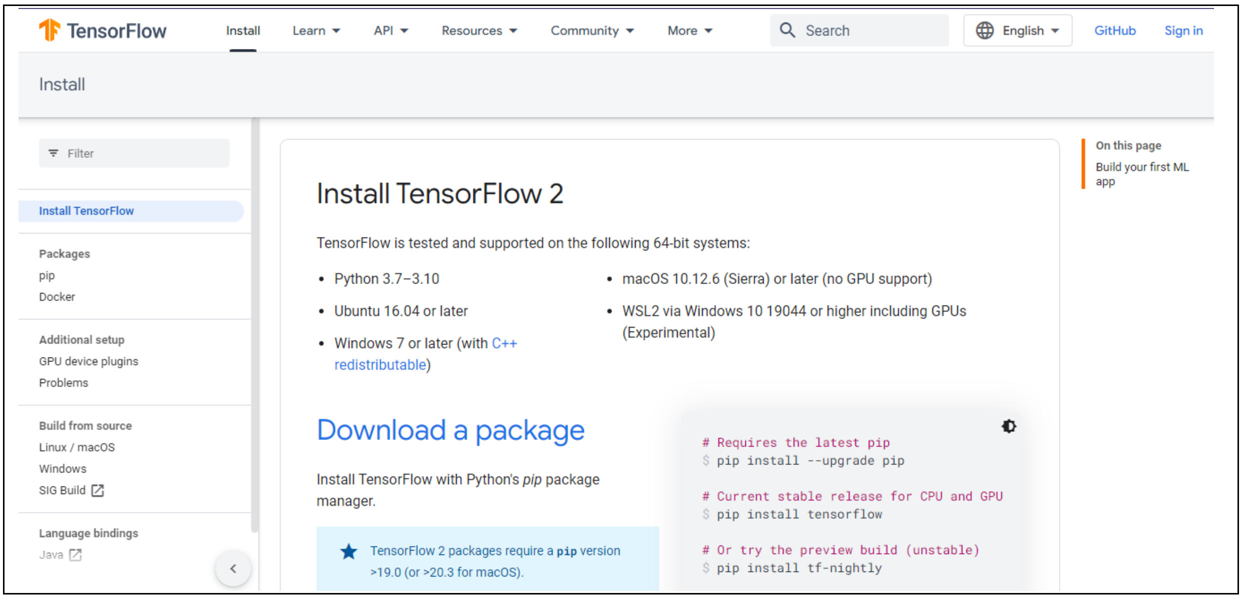
Below is a **code cell**. Once the toolbar button indicates CONNECTED, click in the cell to select it and execute the contents in the following ways:

- Click the **Play icon** in the left gutter of the cell;
- Type **Cmd/Ctrl+Enter** to run the cell in place;
- Type **Shift+Enter** to run the cell and move focus to the next cell (adding one if none exists); or
- Type **Alt+Enter** to run the cell and insert a new code cell immediately below it.

There are additional options for running some or all cells in the **Runtime** menu.

**Figure 2. The Google Colaboratory login window**

- ## TensorFlow Hub



Install TensorFlow 2

TensorFlow is tested and supported on the following 64-bit systems:

- Python 3.7–3.10
- Ubuntu 16.04 or later
- Windows 7 or later (with C++ redistributable)
- macOS 10.12.6 (Sierra) or later (no GPU support)
- WSL2 via Windows 10 19044 or higher including GPUs (Experimental)

Download a package

Install TensorFlow with Python's *pip* package manager.

TensorFlow 2 packages require a **pip** version >19.0 (or >20.3 for macOS).

```
# Requires the latest pip
$ pip install --upgrade pip

# Current stable release for CPU and GPU
$ pip install tensorflow

# Or try the preview build (unstable)
$ pip install tf-nightly
```

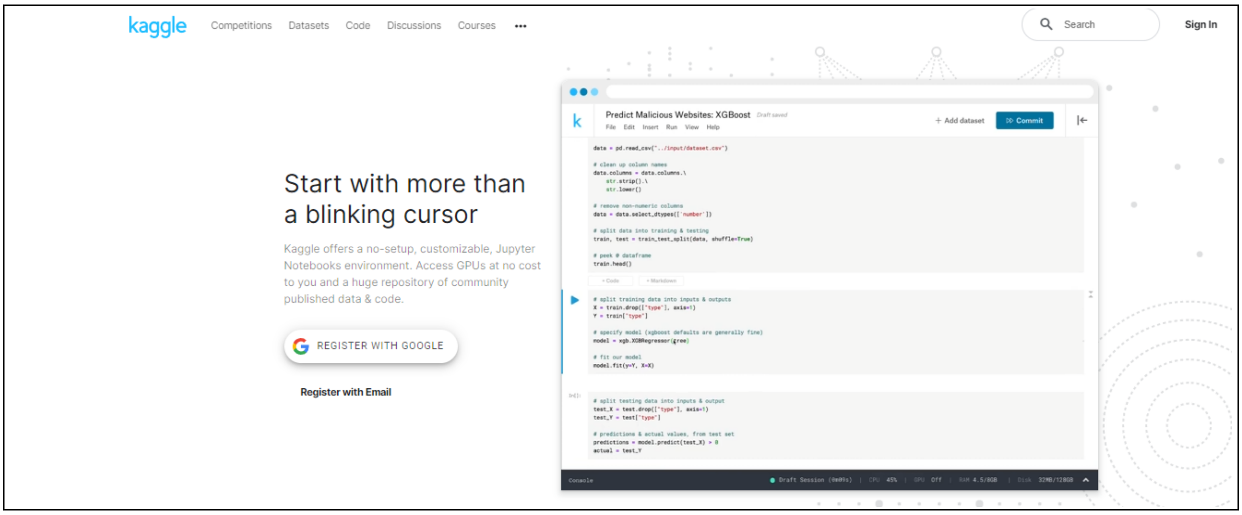
**Figure 3. The TensorFlow hub login page**

The TensorFlow hub is a repository of various packages and libraries required to build the various models in deep learning. In this research, various packages and libraries have been loaded from TensorFlow[3]. For transformer-based models the pre-processor and encoder are mandatory and it has taken in the research from TensorFlow Hub. The Figure 3 represents the sign in page of Tenserflow hub.

- ## Data Source

[3] https://www.tensorflow.org/install

The data source used in this dataset has been taken from the Public repository Kaggle[4]. Since the dataset was available publically there were no ethical concerns. The dataset was having 2 features reviews & sentiments. The dataset was first downloaded from Kaggle and after that, it has been saved to google drive and accessed since it allows faster access and easy processing of data. Figure 4 shows the welcome page of Kaggle. From here various datasets can be accessed.



Start with more than a blinking cursor

Kaggle offers a no-setup, customizable, Jupyter Notebooks environment. Access GPUs at no cost to you and a huge repository of community published data & code.

```
data = pd.read_csv("../input/dataset.csv")

# clean up column names
data.columns = data.columns.\
    str.strip().\
    str.lower()

# remove non-numeric columns
data = data.select_dtypes(['number'])

# split data into training & testing
train, test = train_test_split(data, shuffle=True)

# peek @ dataframe
train.head()
```

```
# split training data into inputs & outputs
X = train.drop(["type"], axis=1)
Y = train["type"]

# specify model (xgboost defaults are generally fine)
model = xgb.XGBRegressor(free)

# fit our model
model.fit(y=Y, X=X)
```

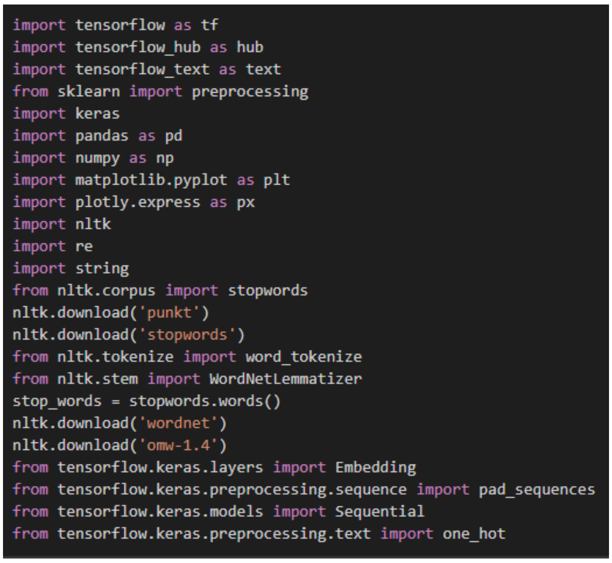
```
# split testing data into inputs & output
test_X = test.drop(["type"], axis=1)
test_Y = test["type"]

# predictions & actual values, from test set
predictions = model.predict(test_X) > 0
actual = test_Y
```

**Figure 4. The Kaggle repository**

# 4 Project Development

In this section, only important steps which are necessary to carry out have been explained. By referring to these steps reader will get a basic understanding of necessary packages & libraries as well.

- **Importing the necessary libraries & Packages**



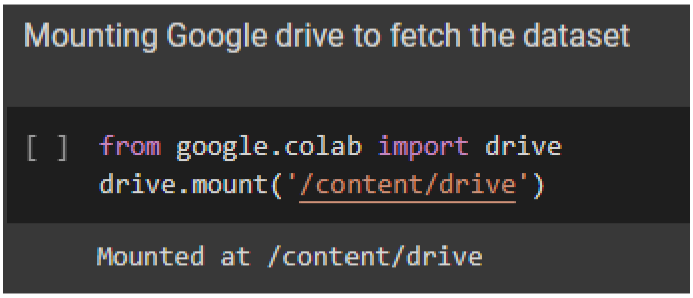
```
import tensorflow as tf
import tensorflow_hub as hub
import tensorflow_text as text
from sklearn import preprocessing
import keras
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import nltk
import re
import string
from nltk.corpus import stopwords
nltk.download('punkt')
nltk.download('stopwords')
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
stop_words = stopwords.words()
nltk.download('wordnet')
nltk.download('omw-1.4')
from tensorflow.keras.layers import Embedding
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.preprocessing.text import one_hot
```

**Figure 5. Importing Packages & Libraries**

[4] https://www.kaggle.com/

From Figure 5 it can be seen that various packages & libraries have been imported. All this packages are required for NLP tasks. Apart from that for modelling various layers can be imported from Tenserflow.

- **Mounting the Google drive:** Figure 6 illustrates how to mount google drive. Here path can be changed depending on the requirements. Once google drive will be mounted all the files from the drive can be accessed easily and also we can store the files back to drive.



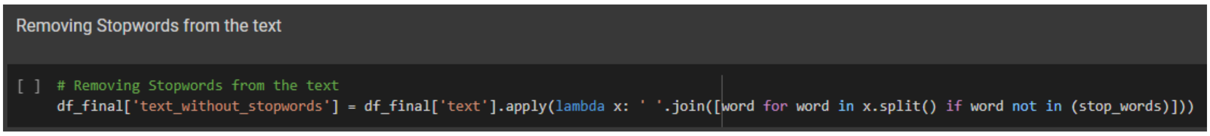
Mounting Google drive to fetch the dataset

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

     Mounted at /content/drive
```

**Figure 6. Mounting Google Drive**

- **Stop words removal**

In every NLP task pre-processing is having very huge importance. If stop words are present in the dataset then it will hinder the models performance since the model will focus more on unimportant words. Figure 7 will illustrate the process of Stopwords removal.



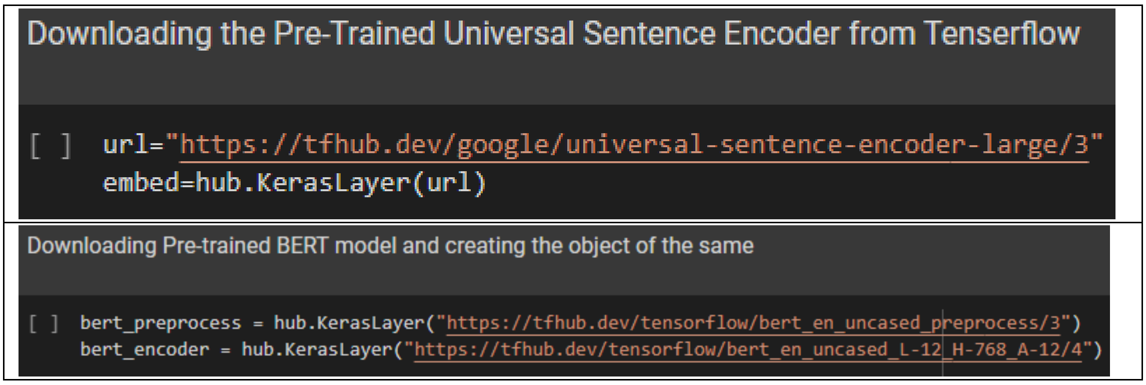
Removing Stopwords from the text

```
[ ]  # Removing Stopwords from the text
     df_final['text_without_stopwords'] = df_final['text'].apply(lambda x: ' '.join([word for word in x.split() if word not in (stop_words)]))
```

**Figure 7. Stop words removal**

- **Downloading the Pre-processor & encoder from TensorFlow Hub**

For Transformer based models the pre-processor & encoder has been required. It can be fetched from the TensorFlow hub and can store in a variable to access later on in the model building. Figure 8. Illustrates the process of downloading the pre-processor & encoder from TensorFlow.



Downloading the Pre-Trained Universal Sentence Encoder from Tenserflow

```
[ ]  url="https://tfhub.dev/google/universal-sentence-encoder-large/3"
     embed=hub.KerasLayer(url)
```

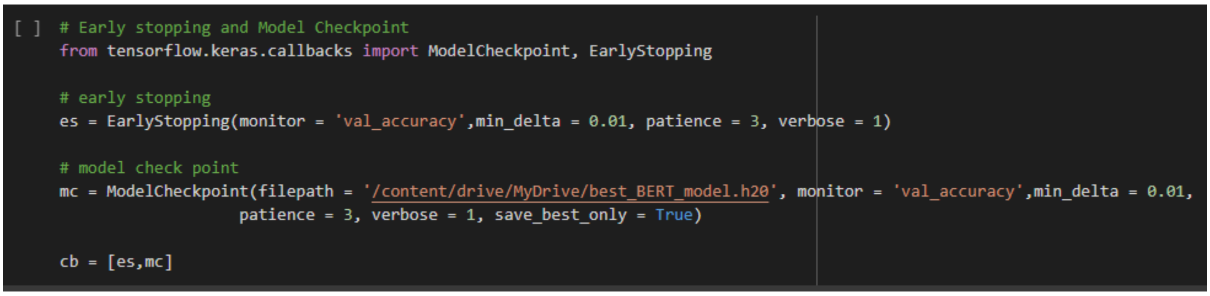
Downloading Pre-trained BERT model and creating the object of the same

```
[ ]  bert_preprocess = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/3")
     bert_encoder = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/4")
```

**Figure 8. Downloading the Pre-processor & Encoder**

- **Use case of Early stopping Mechanism**

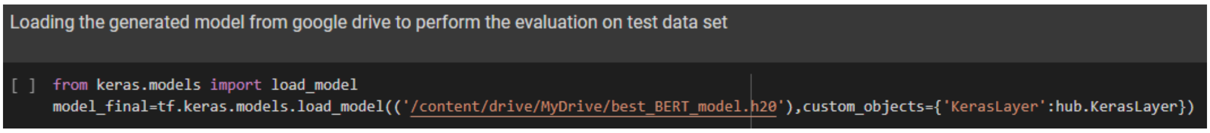
The early stopping has been used in this research to prevent the model from getting overfit. The early stopping will be triggered when the accuracy won't improve by 0.01 value after 3 epochs. The best model will automatically save to google drive at the given path and can easily access and load. Figure 9 illustrates the various parameters used for early stopping.



```
# Early stopping and Model Checkpoint
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping

# early stopping
es = EarlyStopping(monitor = 'val_accuracy',min_delta = 0.01, patience = 3, verbose = 1)

# model check point
mc = ModelCheckpoint(filepath = '/content/drive/MyDrive/best_BERT_model.h20', monitor = 'val_accuracy',min_delta = 0.01,
                     patience = 3, verbose = 1, save_best_only = True)

cb = [es,mc]
```

**Figure 9. Early stopping mechanism**

- **Loading the Generated model from the drive**



Loading the generated model from google drive to perform the evaluation on test data set

```
from keras.models import load_model
model_final=tf.keras.models.load_model(('/content/drive/MyDrive/best_BERT_model.h20'),custom_objects={'KerasLayer':hub.KerasLayer})
```

**Figure 10. Accessing the generated model**

Figure 10 shows how to load & access the generated model from the drive. With the help of the Keras model can be fetched and accessed. Once it is accessible we can pass the testing data to check the predictions.