Importing all the necessary libraries

```
import pandas as pd  # df processing, CSV file I/O (e.g. pd.read_csv)
import numpy as np  # linear algebra
```

```
!wget https://raw.githubusercontent.com/yogawicaksana/helper_prabowo/main/helper_prabowo_ml.py
```

```
--2023-03-04 08:29:52--  https://raw.githubusercontent.com/yogawicaksana/helper_prabowo/main/helper_prabowo_ml.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.109.133, 185.199.108.1
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 13881 (14K) [text/plain]
Saving to: ‘helper_prabowo_ml.py.1’

helper_prabowo_ml.p 100%[===================>]  13.56K  --.-KB/s    in 0s

2023-03-04 08:29:52 (90.4 MB/s) - ‘helper_prabowo_ml.py.1’ saved [13881/13881]
```

```
!pip install transformers   #import transformers
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: transformers in /usr/local/lib/python3.8/dist-packages (4.26.1)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.8/dist-packages (from transformers) (23.0
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.8/dist-packages (from transformers) (20
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.8/dist-packages (from transformers) (1.22.4)
Requirement already satisfied: tokenizers!=0.11.3,<0.14,>=0.11.1 in /usr/local/lib/python3.8/dist-packages (from t
Requirement already satisfied: huggingface-hub<1.0,>=0.11.0 in /usr/local/lib/python3.8/dist-packages (from transf
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.8/dist-packages (from transformers) (4.64.1)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.8/dist-packages (from transformers) (6.0)
Requirement already satisfied: requests in /usr/local/lib/python3.8/dist-packages (from transformers) (2.25.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.8/dist-packages (from transformers) (3.9.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.8/dist-packages (from huggingf
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.8/dist-packages (from requests->tra
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.8/dist-packages (from requests->transformers
Requirement already satisfied: chardet<5,>=3.0.2 in /usr/local/lib/python3.8/dist-packages (from requests->transfo
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.8/dist-packages (from requests->transf
```

```
from helper_prabowo_ml import clean_html, remove_links, non_ascii, lower, email_address, removeStopWords, punct, remove_
import matplotlib.pyplot as plt
import seaborn as sns
import warnings, re
warnings.filterwarnings("ignore")
from sklearn.model_selection import train_test_split
from transformers import AutoTokenizer, TFBertModel
from tensorflow.keras.utils import to_categorical
import tensorflow as tf
from tensorflow.keras.layers import Dense, Input, GlobalMaxPool1D, Dropout
from tensorflow.keras.models import Model
from tensorflow.keras.initializers import TruncatedNormal
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.losses import CategoricalCrossentropy
from tensorflow.keras.metrics import CategoricalAccuracy
from sklearn.metrics import classification_report
from tensorflow.keras.utils import plot_model
```

```
# reading the give data

train = pd.read_csv('/content/train.txt', names = ['Input', 'Sentiment'], sep =';', encoding = 'utf-8')
```

```
val = pd.read_csv("/content/val.txt",names=['Input','Sentiment'],sep=';',encoding='utf-8')
test = pd.read_csv("/content/train.txt",names=['Input','Sentiment'],sep=';',encoding='utf-8')
```

```
df = pd.concat([train, test, val], axis=0)

df.head()
```

	Input	Sentiment
0	i didnt feel humiliated	sadness
1	i can go from feeling so hopeless to so damned...	sadness
2	im grabbing a minute to post i feel greedy wrong	anger
3	i am ever feeling nostalgic about the fireplac...	love
4	i am feeling grouchy	anger

```
df = df.sample(frac=0.1)
df = df.reset_index()
df.head()
```

	index	Input	Sentiment
0	1596	i feel uncomfortable with the fact i am so pow...	fear
1	12393	i wont feel so damn idiotic	sadness
2	6792	i feel ignored and invisible so every weekend ...	sadness
3	12080	i feel like ive been fairly successful	joy
4	9440	i feel like the cute little case is kind of hi...	joy

```
df.drop('index',axis=1,inplace=True)
```

```
df.head()
```

	Input	Sentiment
0	i feel uncomfortable with the fact i am so pow...	fear
1	i wont feel so damn idiotic	sadness
2	i feel ignored and invisible so every weekend ...	sadness
3	i feel like ive been fairly successful	joy
4	i feel like the cute little case is kind of hi...	joy

```
df.shape
```

```
(3400, 2)
```

Text Preprocessing

```
def preprocess_data(data,col):
    data[col] = data[col].apply(func=clean_html)
    data[col] = data[col].apply(func=remove_digits)
    data[col] = data[col].apply(func=remove_)
    data[col] = data[col].apply(func=removeStopWords)
    data[col] = data[col].apply(func=remove_links)
    data[col] = data[col].apply(func=remove_special_characters)
    data[col] = data[col].apply(func=non_ascii)
    data[col] = data[col].apply(func=email_address)
    data[col] = data[col].apply(func=punct)
    data[col] = data[col].apply(func=lower)
    return data
```

```
df['Input']
```

```
0       i feel uncomfortable with the fact i am so pow...
1                             i wont feel so damn idiotic
2       i feel ignored and invisible so every weekend ...
3                      i feel like ive been fairly successful
4       i feel like the cute little case is kind of hi...
                              ...
3395    i done something that i didn t feel inspired o...
3396    ive been feeling weirdly superior about my kno...
3397    i feel so repressed when compared to dear a hr...
3398    i started the third block feeling hot and cold...
3399    i remember feeling humiliated because of the p...
Name: Input, Length: 3400, dtype: object
```

```
df['Sentiment'].value_counts()
```

```
joy         1145
sadness      973
anger        467
fear         412
love         271
surprise     132
Name: Sentiment, dtype: int64
```

```
preprocessed_df = preprocess_data(df,'Input')
preprocessed_df.head()
```

	Input	Sentiment
0	feel uncomfortable fact powerless moment	fear
1	wont feel damn idiotic	sadness
2	feel ignored invisible every weekend miserable	sadness
3	feel ive fairly successfu	joy
4	feel cute little case kind hidde	joy

```
preprocessed_df['num_words'] = preprocessed_df.Input.apply(len)
```

```
preprocessed_df.head()
```

	Input	Sentiment	num_words
0	feel uncomfortable fact powerless moment	fear	40
1	wont feel damn idiotic	sadness	22
2	feel ignored invisible every weekend miserable	sadness	46
3	feel ive fairly successfu	joy	25
4	feel cute little case kind hidde	joy	32

```
encoded_labels = {'anger': 0, 'fear': 1, 'joy': 2, 'love': 3, 'sadness': 4, 'surprise': 5}
```

```
preprocessed_df.head()
```

	Input	Sentiment	num_words
0	feel uncomfortable fact powerless moment	fear	40
1	wont feel damn idiotic	sadness	22
2	feel ignored invisible every weekend miserable	sadness	46
3	feel ive fairly successfu	joy	25
4	feel cute little case kind hidde	joy	32

Train-Test Split

```
train_data, test_data = train_test_split(preprocessed_df,test_size=0.3,random_state=101,shuffle=True,stratify=preprocess
```

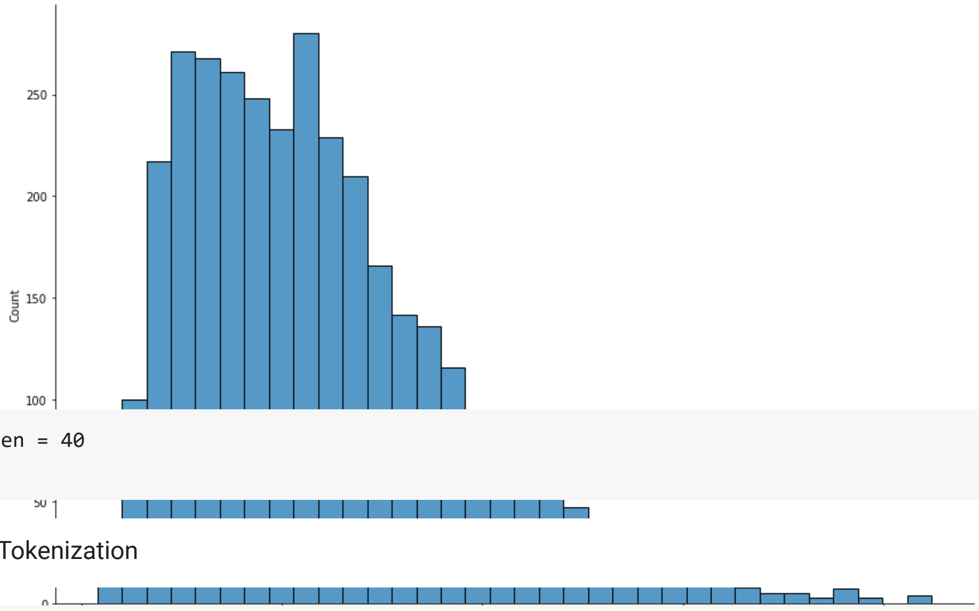
Loading the Tokenizer class and pretrained BERT model¶

```
tokenizer = AutoTokenizer.from_pretrained("bert-base-uncased")
bert_model = TFBertModel.from_pretrained("bert-base-uncased")
```

```
Downloading                                         28.0/28.0 [00:00<00:00,
(…)okenizer_config.json: 100%                       397B/s]
Downloading                                         570/570 [00:00<00:00,
(…)lve/main/config.json: 100%                       10.3kB/s]
Downloading                                         232k/232k [00:00<00:00,
(…)solve/main/vocab.txt: 100%                       1.89MB/s]
Downloading                                         466k/466k [00:00<00:00,
(…)/main/tokenizer.json: 100%                       3.37MB/s]
Downloading (…)"tf_model.h5";:                      536M/536M [00:05<00:00,
```

```
sns.displot(preprocessed_df.num_words,height=8,aspect=1.5)
```

```
<seaborn.axisgrid.FacetGrid at 0x7ff0617034c0>
```



```
max_len = 40
```

Text Tokenization

```
x_train = tokenizer(text=train_data.Input.tolist(),
                   add_special_tokens=True,
                   return_tensors='tf',
                   max_length=max_len,
                   padding=True,
                   truncation=True,
                   return_token_type_ids=False,
                   return_attention_mask=True,
                   verbose=True
                   )

x_test = tokenizer(text=test_data.Input.tolist(),
                   add_special_tokens=True,
                   return_tensors='tf',
                   max_length=max_len,
                   padding=True,
                   truncation=True,
                   return_token_type_ids=False,
                   return_attention_mask=True,
                   verbose=True
                  )
```

Defining the model architecture

```
input_ids = Input(shape=(max_len,),name='input_ids',dtype=tf.int32)
attention_mask = Input(shape=(max_len,),name='attention_mask',dtype=tf.int32)
```

```
input_ids
```

```
<KerasTensor: shape=(None, 40) dtype=int32 (created by layer 'input_ids')>
```

```
attention_mask
```

```
<KerasTensor: shape=(None, 40) dtype=int32 (created by layer 'attention_mask')>
```

```
embeddings = bert_model(input_ids,attention_mask=attention_mask)[0] # 0: final hidden state, 1: pooling output
output = GlobalMaxPool1D()(embeddings)
output = Dense(units=128,activation='relu')(output)
output = Dropout(0.1)(output)
output = Dense(units=64,activation='relu')(output)
output = Dense(units=32,activation='relu')(output)
y = Dense(units=6,activation='softmax')(output)

model = Model(inputs=[input_ids,attention_mask],outputs=y)
model.layers[2].trainable = True
```

Compiling the model

```
model.compile(loss=CategoricalCrossentropy(from_logits=True),
              optimizer=tf.keras.optimizers.legacy.Adam(learning_rate=5e-5,epsilon=1e-8,decay=0.01,clipnorm=1.0),
              metrics=CategoricalAccuracy('balanced_accuracy'))
```

Encoding the emotion labels

```
train_data['Label'] = train_data.Sentiment.map(encoded_labels)
test_data['Label'] = test_data.Sentiment.map(encoded_labels)
```

```
train_data['Label']
```

```
649     0
2707    2
56      1
1836    2
2976    2
       ..
2993    0
2666    4
3054    4
1542    2
59      4
Name: Label, Length: 2380, dtype: int64
```

```
train_data.head()
```

	Input	Sentiment	num_words	Label
649	mean feel always someone else people becuase w...	anger	75	0
2707	think writing fun fulfilling think decide intr...	joy	89	2
56	feel strange weird entire struggle one deals k...	fear	58	1
1836	tested tried feel confident making bold statem...	joy	78	2
2976	spent days problem feeling eager finish job pl...	joy	128	2

```
test_data.head()
```

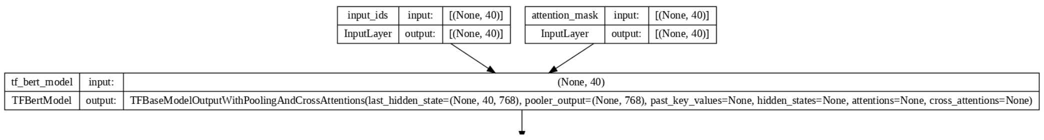
	Input	Sentiment	num_words	Label
246	otice worrying push feeling away replace thoug...	joy	88	2
2944	began feel agitated wanted buy ewan food medic...	anger	54	0
1193	definitely feel appreciative boyfriend	joy	38	2
2302	originals want rebekah satisfactory ending sha...	love	89	3
221	eaving feel inadaquate valued appreciated	joy	41	2

Generating the model summary and plot

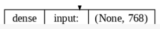
```
model.summary()
```

```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to                     
==================================================================================================
 input_ids (InputLayer)         [(None, 40)]         0           []                               
                                                                                                  
 attention_mask (InputLayer)    [(None, 40)]         0           []                               
                                                                                                  
 tf_bert_model (TFBertModel)    TFBaseModelOutputWi  109482240   ['input_ids[0][0]',              
                                thPoolingAndCrossAt               'attention_mask[0][0]']         
                                tentions(last_hidde                                               
                                n_state=(None, 40,                                                
                                768),                                                             
                                 pooler_output=(Non                                               
                                e, 768),                                                          
                                 past_key_values=No                                               
                                ne, hidden_states=N                                               
                                one, attentions=Non                                               
                                e, cross_attentions                                               
                                =None)                                                            
                                                                                                  
 global_max_pooling1d (GlobalMa  (None, 768)         0           ['tf_bert_model[0][0]']          
 xPooling1D)                                                                                      
                                                                                                  
 dense (Dense)                  (None, 128)          98432       ['global_max_pooling1d[0][0]']   
                                                                                                  
 dropout_37 (Dropout)           (None, 128)          0           ['dense[0][0]']                  
                                                                                                  
 dense_1 (Dense)                (None, 64)           8256        ['dropout_37[0][0]']             
                                                                                                  
 dense_2 (Dense)                (None, 32)           2080        ['dense_1[0][0]']                
                                                                                                  
 dense_3 (Dense)                (None, 6)            198         ['dense_2[0][0]']                
                                                                                                  
==================================================================================================
Total params: 109,591,206
Trainable params: 109,591,206
Non-trainable params: 0
__________________________________________________________________________________________________
```

```
plot_model(model,'model.png',show_shapes=True,dpi=100)
```


input_ids
InputLayer
input:
output:
[(None, 40)]
[(None, 40)]
attention_mask
InputLayer
input:
output:
[(None, 40)]
[(None, 40)]
tf_bert_model
TFBertModel
input:
output:
(None, 40)
TFBaseModelOutputWithPoolingAndCrossAttentions(last_hidden_state=(None, 40, 768), pooler_output=(None, 768), past_key_values=None, hidden_states=None, attentions=None, cross_attentions=None)

Training and fine-tuning the pretrained BERT model

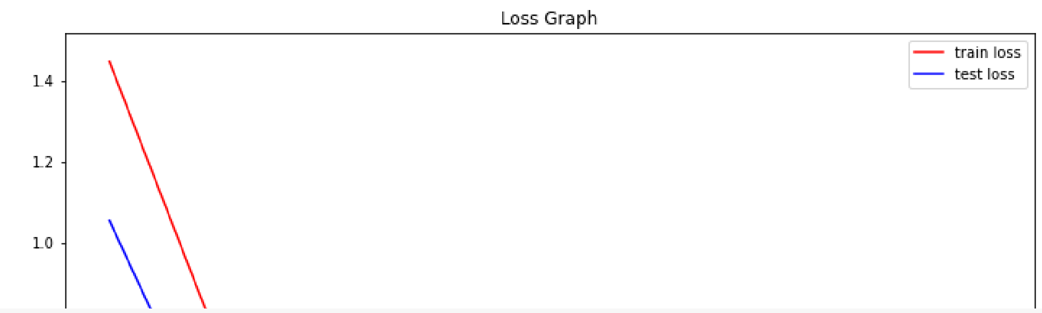


```
r = model.fit(x={'input_ids': x_train['input_ids'], 'attention_mask': x_train['attention_mask']},
             y=to_categorical(train_data.Label),
             epochs=10,
             batch_size=32,
             validation_data=({'input_ids': x_test['input_ids'], 'attention_mask': x_test['attention_mask']},to_categori
```

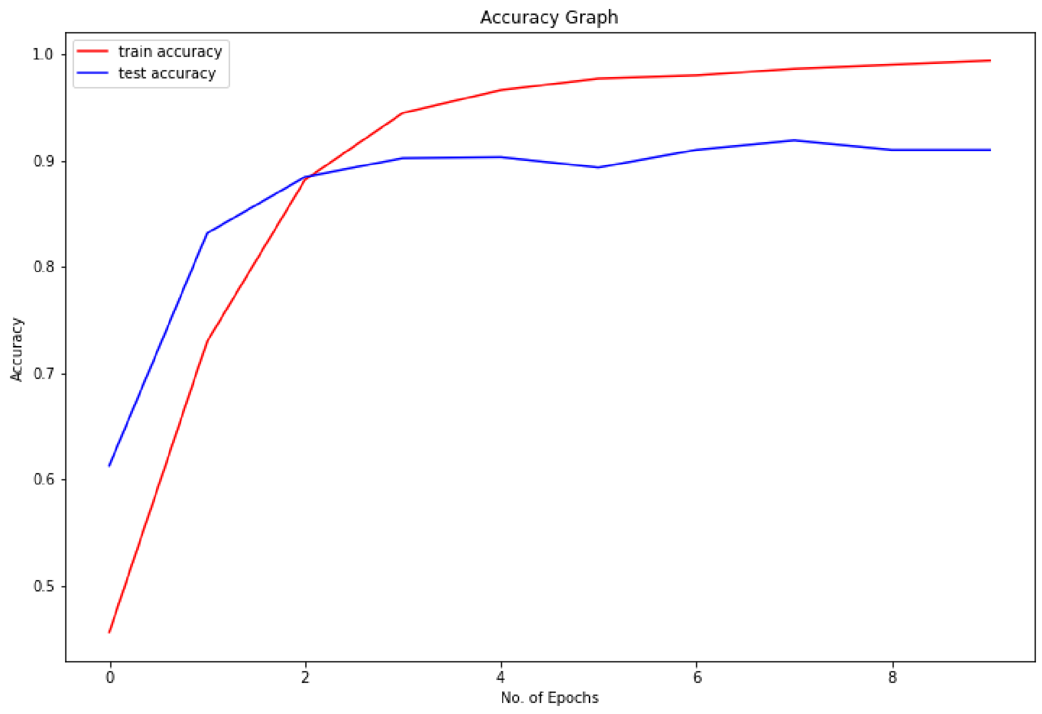
```
Epoch 1/10
WARNING:tensorflow:Gradients do not exist for variables ['tf_bert_model/bert/pooler/dense/kernel:0', 'tf_bert_mode
WARNING:tensorflow:Gradients do not exist for variables ['tf_bert_model/bert/pooler/dense/kernel:0', 'tf_bert_mode
75/75 [==============================] - 1448s 18s/step - loss: 1.4460 - balanced_accuracy: 0.4559 - val_loss: 1.0
Epoch 2/10
75/75 [==============================] - 1339s 18s/step - loss: 0.8235 - balanced_accuracy: 0.7294 - val_loss: 0.5
Epoch 3/10
75/75 [==============================] - 1336s 18s/step - loss: 0.3960 - balanced_accuracy: 0.8815 - val_loss: 0.3
Epoch 4/10
75/75 [==============================] - 1328s 18s/step - loss: 0.2089 - balanced_accuracy: 0.9445 - val_loss: 0.3
Epoch 5/10
75/75 [==============================] - 1322s 18s/step - loss: 0.1309 - balanced_accuracy: 0.9660 - val_loss: 0.3
Epoch 6/10
75/75 [==============================] - 1265s 17s/step - loss: 0.0903 - balanced_accuracy: 0.9769 - val_loss: 0.4
Epoch 7/10
75/75 [==============================] - 1273s 17s/step - loss: 0.0707 - balanced_accuracy: 0.9798 - val_loss: 0.3
Epoch 8/10
75/75 [==============================] - 1444s 19s/step - loss: 0.0537 - balanced_accuracy: 0.9861 - val_loss: 0.3
Epoch 9/10
75/75 [==============================] - 1429s 19s/step - loss: 0.0444 - balanced_accuracy: 0.9899 - val_loss: 0.3
Epoch 10/10
75/75 [==============================] - 1327s 18s/step - loss: 0.0355 - balanced_accuracy: 0.9937 - val_loss: 0.3
```

Analyzing model performance

```
plt.figure(figsize=(12,8))
plt.plot(r.history['loss'],'r',label='train loss')
plt.plot(r.history['val_loss'],'b',label='test loss')
plt.xlabel('No. of Epochs')
plt.ylabel('Loss')
plt.title('Loss Graph')
plt.legend();
```



```
plt.figure(figsize=(12,8))
plt.plot(r.history['balanced_accuracy'],'r',label='train accuracy')
plt.plot(r.history['val_balanced_accuracy'],'b',label='test accuracy')
plt.xlabel('No. of Epochs')
plt.ylabel('Accuracy')
plt.title('Accuracy Graph')
plt.legend();
```



Saving the model

```
model.save("emotion_detector.h5")
```

Evaluating the model on the test dataset

```
loss, acc = model.evaluate({'input_ids': x_test['input_ids'], 'attention_mask': x_test['attention_mask']},to_categorical
print("Test Categorical Cross-Entropy Loss:",loss)
print("Test Categorical Accuracy:",acc)
```

```
32/32 [==============================] - 177s 5s/step - loss: 0.3738 - balanced_accuracy: 0.9098
Test Categorical Cross-Entropy Loss: 0.3738430440425873
Test Categorical Accuracy: 0.9098039269447327
```

```
test_predictions = model.predict({'input_ids': x_test['input_ids'], 'attention_mask': x_test['attention_mask']})
test_predictions = np.argmax(test_predictions,axis=1)
print(classification_report(test_data.Label,test_predictions))
```

```
32/32 [==============================] - 169s 5s/step
              precision    recall  f1-score   support

           0       0.91      0.91      0.91       140
           1       0.90      0.90      0.90       124
           2       0.93      0.92      0.93       343
           3       0.82      0.86      0.84        81
           4       0.95      0.91      0.93       292
           5       0.71      0.88      0.79        40

    accuracy                           0.91      1020
   macro avg       0.87      0.90      0.88      1020
weighted avg       0.91      0.91      0.91      1020
```

✓ 2m 49s completed at 6:40 PM