BERT Classification Modeling -KERAS

@Author : Woojin Park, Nidhi Bhaskar

@Email : woojinpa@andrew.cmu.edu , nidhibha@andrew.cmu.edu

@Status : In-Progress

@article{turc2019,title={Well-Read Students Learn Better: On the Importance of Pre-training Compact Models}, author={Turc, Iulia and Chang, Ming-Wei and Lee, Kenton and Toutanova, Kristina}, journal= {arXiv preprintarXiv:1908.08962v2 },year={2019}}

In [2]:

```
### Import Relevant Libraries
import os
import pandas as pd
import numpy as np
import collections
import datetime as dt
import requests
import json
import re
import time

import matplotlib.pyplot as plt
import matplotlib.cm as cm
%matplotlib inline
import seaborn as sns
sns.set_style('whitegrid')
from scipy.stats import norm
from IPython.display import display, Image

import string
import re
import nltk
from nltk.util import ngrams
from nltk import pos_tag,word_tokenize
from nltk.corpus import stopwords
from nltk.tokenize import WhitespaceTokenizer
from nltk.stem import WordNetLemmatizer,PorterStemmer
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
from textblob import TextBlob

from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
from sklearn import metrics
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation as LDA
from sklearn.decomposition import NMF
from sklearn.model_selection import train_test_split
```

In [3]:

```
### Build a get_date function to convert date format
#### Build a data_creation function to read json data into pandas dataframe

def get_date(created):
    return dt.datetime.fromtimestamp(created)

def data_creation(subreddit) :
    with open('submissions_'+subreddit+'.json') as f:
        data = json.loads("[" +
            f.read().replace("}\n{", "},\n{") +
        "]")
    data =pd.DataFrame(data)
    reddit_data = data[['author','over_18','title','selftext','num_comments', 's
core', 'full_link','created_utc']]
    reddit_data = reddit_data.dropna()
    _timestamp = reddit_data["created_utc"].apply(get_date)
    reddit_data = reddit_data.assign(timestamp = _timestamp)
    reddit_data['over_18'] = reddit_data['over_18'].astype('str')
    reddit_data['subreddit']= subreddit
    # Build column have title + selftext
    reddit_data['title_with_selftext']= reddit_data['title'] +" " + reddit_data[
'selftext']

    # Do one more extra cleaning : keep updating this part
    reddit_data=reddit_data[~reddit_data['title_with_selftext'].isin([ '[removed
]', '[deleted]',''])]
    subreddit = reddit_data
    return subreddit

def empty_words_clean(text):
    text = text.replace('[removed]','')
    text= text.replace('[deleted]','')
    text= text.replace('\n','')
    return (text)
```

In [4]:

```
pd.set_option('display.max_columns', 500)
pd.set_option('display.max_rows', 500)
```

In [5]:

```
#### 0: SuicideWatch 0.8
#### 1 : depressed
#### 2 : happy 0.9
#### 3 :  selfimprovement 0.9



### Dataframing 4 subreddit Datasets
SuicideWatch_df = data_creation('SuicideWatch')
depressed_df = data_creation('depressed')
happy_df = data_creation('happy')
selfimprovement_df = data_creation('selfimprovement')


SuicideWatch_df = SuicideWatch_df.sample(frac=0.8, replace=True, random_state=1)
depressed_df = depressed_df.sample(frac=0.99, replace=True, random_state=1)
happy_df = happy_df.sample(frac=0.9, replace=True, random_state=1)
selfimprovement_df = selfimprovement_df.sample(frac=0.9, replace=True, random_st
ate=1)



### Concat all 4 dataframes into one merged file
all_subreddit_df = pd.concat([SuicideWatch_df,depressed_df,happy_df,selfimprovem
ent_df])
```

In [6]:

```
### 0.Data Preparation
```

In [7]:

```
#### Text Preprocessing by following pipeline :
### Raw text => Tokeninze/lowercase => Remove stop words => Remove non-alphabeti
c characters =>
### Remove Extra Punctuations => Lemmatization => Build Custom Stop words dictio
nary
```

In [8]:

```
# Build function that takes a word and returns true if it consists only of non-a
lphabetic characters
def alpha_filter(w):
    pattern = re.compile('^[^a-z]+$')
    if (pattern.match(w)):
        return True
    else:
        return False

# Build data preparation function including all the necessary 7 steps :
def clean_words(text):
    # lower text & tokenizing
    text = text.replace('\n',' ')
    text = text.replace('[removed]',' ')
    text= text.replace('[deleted]', '')
    text =text.lower()

    ### Updated cleaning-pipeline :
    text = re.sub(r'[^a-zA-Z0-9 ]',r' ',text) #remove anything that is not a let
ter or number first
    text = [word for word in text.split(" ")]

    # remove stop words
    nltk_stopwords = set(stopwords.words('english'))
    review_lower_stop = [x for x in text if not x in nltk_stopwords]
    # remove extra punctuations
    review_lower_stop_pun = [y for y in review_lower_stop if not alpha_filter(y)
]
    review_lower_stop_pun_extra = [''.join(x for x in par if x not in string.pun
ctuation) for par in review_lower_stop_pun]
    # Lemmatization
    porter = WordNetLemmatizer()
    review_lower_stop_pun_extra_lemmatized = []
    for a in review_lower_stop_pun_extra :
        review_lower_stop_pun_extra_lemmatized.append(porter.lemmatize(a))
    # buid custom stop words dictionary
    cachedStopWords = set(stopwords.words("english"))

    ####Keep Updating custom stop words
    cachedStopWords.update(('nt', 'wo', 're', 'im', 'yall','u','ca','ive', 'wan'
,'na','gon','nov','x200b','amp',\
                            'wwwyoutubecomwatch','http','vbjkbl5olvm8','lt', 'br', '
gt', 'amp','tsp','tbsp','nbsp'))
    review_lower_stop_pun_extra_lemmatized_stop = [x for x in review_lower_stop_
pun_extra_lemmatized\
                                                   if not x in cachedStopWords]
    text = " ".join(review_lower_stop_pun_extra_lemmatized_stop)
    return (text)

def detect_polarity(text):
    return TextBlob(text).sentiment.polarity
```

In [9]:

```
##Data Sampling
```

In [10]:

```
### Because of relatively huge dataset, we need to perform random sampling of 30
% for now

all_subreddit_df_list = all_subreddit_df.sample(frac=0.3, replace=True, random_s
tate=1)
```

In [11]:

```
all_subreddit_df_list["title_with_selftext_clean"] = all_subreddit_df_list["titl
e_with_selftext"].apply(lambda x: clean_words(x))
```

In [12]:

```
all_subreddit_df_list['polarity'] = all_subreddit_df_list['title_with_selftext_c
lean'].apply(detect_polarity)
```

In [13]:

```
all_subreddit_df_list.head(3)
```

Out[13]:

	author	over_18	title	selftext	num_comments	score	
12970	sudrawkid	False	Can't properly stick up for myself and feel weak.	Hey everyone! I guess I should just outright g...	2	4	https://ww
5192	PinkyIsSnug	False	Gonna kill myself very very soon. I've really ...	I joined this school with high hopes. hopes th...	5	1	https://ww
2357	ReasonableBrother3	False	GET RID OF DEPRESSION	[removed]	0	1	https://ww

In [14]:

```
#Descriptive statistics & dataframe  info
```

In [15]:

```
all_subreddit_df_list.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 47828 entries, 12970 to 43513
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype         
---  ------                    --------------  -----         
 0   author                    47828 non-null  object        
 1   over_18                   47828 non-null  object        
 2   title                     47828 non-null  object        
 3   selftext                  47828 non-null  object        
 4   num_comments              47828 non-null  int64         
 5   score                     47828 non-null  int64         
 6   full_link                 47828 non-null  object        
 7   created_utc               47828 non-null  int64         
 8   timestamp                 47828 non-null  datetime64[ns]
 9   subreddit                 47828 non-null  object        
 10  title_with_selftext       47828 non-null  object        
 11  title_with_selftext_clean  47828 non-null  object        
 12  polarity                  47828 non-null  float64       
dtypes: datetime64[ns](1), float64(1), int64(3), object(8)
memory usage: 5.1+ MB
```

In [16]:

```
all_subreddit_df_list.describe()
```

Out[16]:

	num_comments	score	created_utc	polarity
count	47828.000000	47828.000000	4.782800e+04	47828.000000
mean	6.214289	65.042590	1.557714e+09	0.076999
std	28.489875	966.074055	3.417972e+07	0.254898
min	0.000000	0.000000	1.304697e+09	-1.000000
25%	1.000000	1.000000	1.538144e+09	-0.038462
50%	2.000000	1.000000	1.570949e+09	0.039981
75%	5.000000	3.000000	1.584263e+09	0.194731
max	1825.000000	81714.000000	1.590173e+09	1.000000

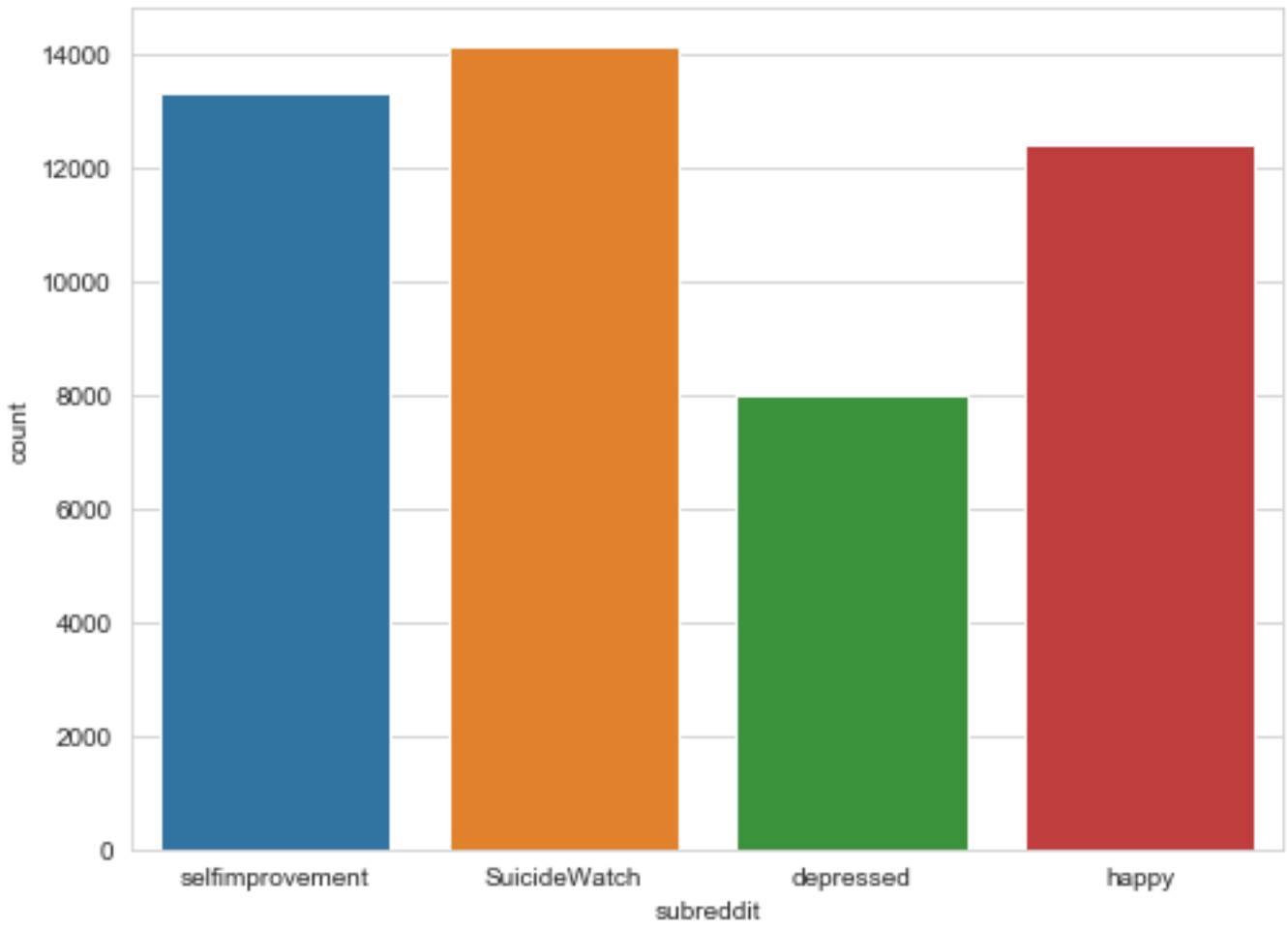
In [17]:

```
# Build Bar plot for all 4 subreddit distribution

plt.figure(figsize=(8,6))
sns.countplot(x='subreddit',data=all_subreddit_df_list)
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a359346d0>
```

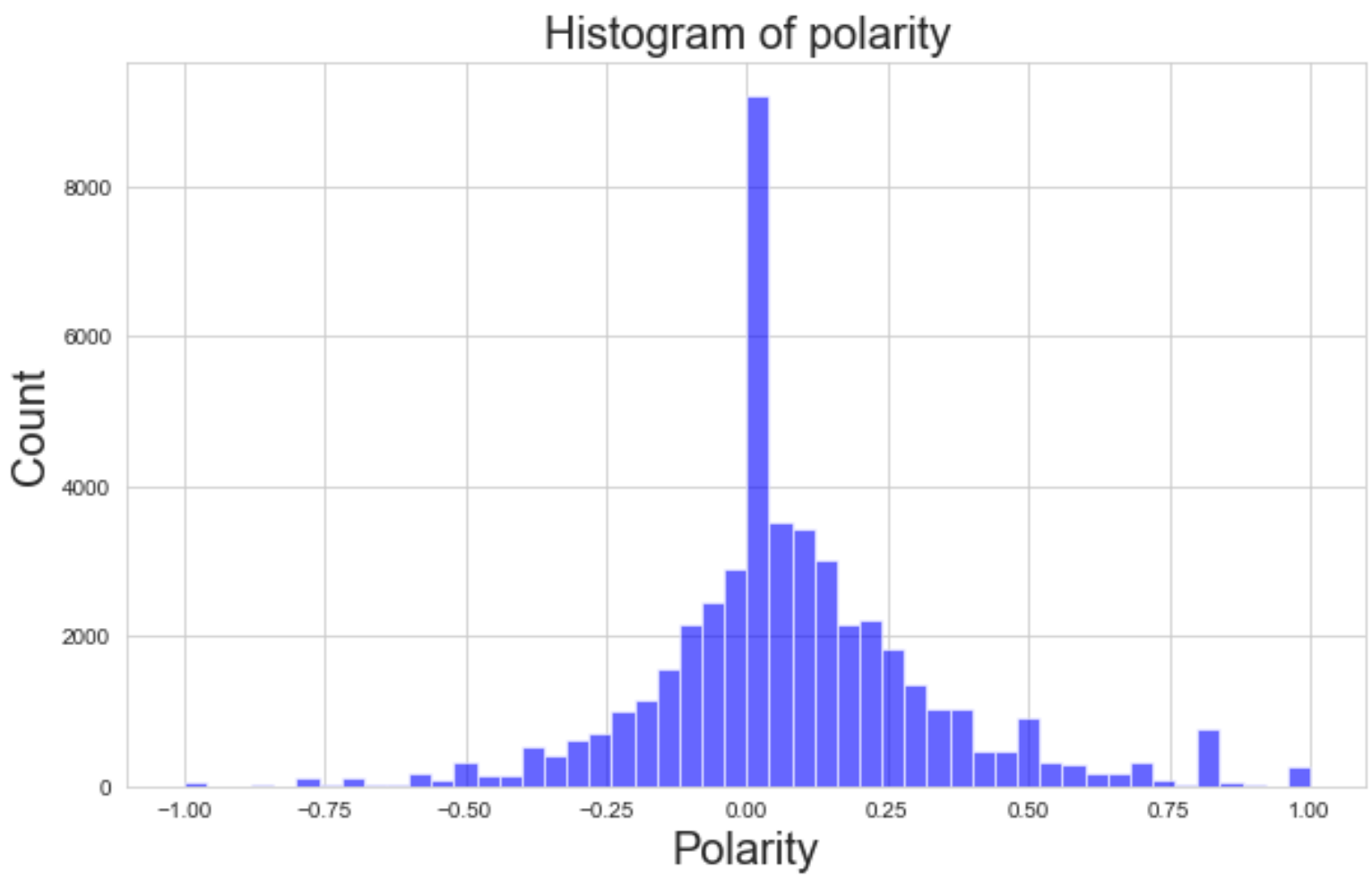


In [18]:

```
## Polarity (Sentiment) distribution of all 4 subreddit - Symmetric/ not skewed
```

In [19]:

```
num_bins = 50
plt.figure(figsize=(10,6))
n, bins, patches = plt.hist(all_subreddit_df_list.polarity, num_bins, facecolor=
'blue', alpha=0.6)
plt.xlabel('Polarity',fontsize =20)
plt.ylabel('Count',fontsize =20)
plt.title('Histogram of polarity',fontsize =20)
plt.show()
```



In [20]:

```
### Build NMF Model
```

In [21]:

```
# get topics with their terms and weights
def get_topics_terms_weights(weights, feature_names):
    feature_names = np.array(feature_names)
    sorted_indices = np.array([list(row[::-1]) for row in np.argsort(np.abs(weig
hts))])
    sorted_weights = np.array([list(wt[index]) for wt, index in zip(weights, sor
ted_indices)])
    sorted_terms = np.array([list(feature_names[row]) for row in sorted_indices]
)
    topics = [np.vstack((terms.T, term_weights.T)).T for terms, term_weights in
zip(sorted_terms, sorted_weights)]
    return topics

# prints components of all the topics  obtained from topic modeling
def print_topics_udf(topics, total_topics=4,
                     weight_threshold=0.0001,
                     display_weights=False,
                     num_terms=None):

    for index in range(total_topics):
        topic = topics[index]
        topic = [(term, float(wt))
                 for term, wt in topic]
        #print(topic)
        topic = [(word, round(wt,2))
                 for word, wt in topic
                 if abs(wt) >= weight_threshold]

        if display_weights:
            print('Topic'+str(index)+' with weights : ')
            print(topic[:num_terms]) if num_terms else topic
            print('\n')
        else:
            print('Topic #'+str(index)+' without weights : ')
            tw = [term for term, wt in topic]
            print(tw[:num_terms]) if num_terms else tw
```

In [22]:

```
def tfidf_nmf_function (dataframe, no_top_words, number) :
# Store only text contents
    data_text = dataframe[['title_with_selftext_clean']]
    data_text['index'] = dataframe.index
    # Assign to 'documents' which has texts and index of each
    documents = data_text
    ## Vectorization
    # NMF is able to use tf-idf - vectorize the corpus
    tfidf_vectorizer = TfidfVectorizer(max_df=0.80, min_df=10, max_features=6000
,ngram_range=(1, 2)
                                       ,stop_words='english', token_pattern = r'\b[^
\d\W]+\b')
```

```
    ## min_df = 10 : ignore words that appear in less than 10 of the subreddits
    ##  max_df=0.80 : model to ignore words that appear in more than 80% of the
subreddits
    ## max_features=6000 : After processing we have a little over 9k(9688) uniqu
e words
    ### so we'll set the max_features to only include the top 6k
    #### by term frequency across the articles for further feature reduction.
    ##  ngram_range=(1, 2) : tf-idf weights for n-grams (bigrams, trigrams etc.)
.
    ### To do that we'll set the n_gram range to (1, 2)
    #### which will include unigrams and bigrams.


    # calculate the feature matrix
    tfidf = tfidf_vectorizer.fit_transform(dataframe['title_with_selftext_clean'
])
    print( "Created %d X %d TF-IDF-normalized document-term matrix" % (tfidf.sha
pe[0], tfidf.shape[1]) )
    tfidf_feature_names = tfidf_vectorizer.get_feature_names()


    print ( "in the corpus of N documents, total of N unique features :")
    display(tfidf.shape)
    tfidf_feature_names = tfidf_vectorizer.get_feature_names()
    print("Length of unique features are : ", len(tfidf_vectorizer.get_feature_n
ames()))

    # Run NMF
    nmf = NMF(n_components=number, random_state=1, alpha=.3, l1_ratio=.5, init='
nndsvd')

    ### Regularization ?
    ####will lower the variance from the model - More robust decision on data as
it minimize overfitting
    ##  'nndsvd'  :Nonnegative Double Singular Value Decomposition which works b
est on sparse data like we have here
    ##  As in ElasticNet, we control the combination of L1 (Lasso) and L2 (Ridge
) with the l1_ratio (ρ) parameter
    ###between 0< - <1  and the intensity of the regularization with the alpha (
α) parameter

    nmf_z = nmf.fit_transform(tfidf)
    nmf_weights = nmf.components_
    nmf_feature_names = tfidf_vectorizer.get_feature_names()


    for topic_idx, topic in enumerate(nmf.components_):
            print("Topic %d:" % (topic_idx))
            print(", ".join([tfidf_feature_names[i]
                        for i in topic.argsort()[:-no_top_words - 1:-1]]) ,"\n")

    topics = get_topics_terms_weights(nmf_weights, nmf_feature_names)
```

```
    print_topics_udf(topics, total_topics=4, num_terms=30, display_weights=True)


    TopicNumber=[]
    for i in range(len(nmf_z)):
        h=nmf_z[i].tolist().index(nmf_z[i].max())
        TopicNumber.append(h)
    documents['topic_nmf']=TopicNumber
    dataframe['topic_nmf'] =TopicNumber
    sns.countplot(x='topic_nmf', data=documents)
```

In [23]:

```
### We are having better results with NMF (over TF-IDF matrix) than with LDA.
### The top keywords of the topics NFM finds are more related and meaningful to
the context of my corpus,
###which are posts of many subjects shared internally in my organization.
```

In [24]:

```
#def nmf_function (tfidf, model, feature_names, no_top_words) :
#nmf_function (tfidf,nmf, tfidf_feature_names, 20)
#visualization topic distribution
tfidf_nmf_function(all_subreddit_df_list,30 ,4)
```

```
/Users/Jay/opt/anaconda3/lib/python3.7/site-packages/ipykernel_launc
her.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pand
as-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py
  after removing the cwd from sys.path.

Created 47828 X 6000 TF-IDF-normalized document-term matrix
in the corpus of N documents, total of N unique features :

(47828, 6000)

Length of unique features are :  6000
Topic 0:
life, time, year, friend, know, people, thing, day, really, think, l
ike, help, going, work, make, got, good, school, love, job, need, th
ought, way, better, self, month, say, lot, family, person

Topic 1:
happy, make happy, make, today, finally, feel happy, year, day, love
, birthday, happy life, dog, really happy, happy happy, want happy,
smile, family, happiness, happy birthday, christmas, picture, little
, video, place, happy new, new, happy today, got, baby, girl

Topic 2:
```

feel, like, feel like, feeling, know, make feel, really, make, peopl
e, dont, sad, anymore, know feel, want, depressed, like feel, thing,
like shit, shit, hate, want feel, good, bad, life feel, talk, think,
felt, feeling like, better, feel better

Topic 3:
want, die, want die, kill, fucking, anymore, dont, hate, live, know,
end, life, tired, pain, want kill, wish, care, dont want, hurt, fuck
, want live, wanna, alive, suicide, want end, shit, stop, death, die
want, scared

Topic0 with weights :
[('life', 2.68), ('time', 2.48), ('year', 2.34), ('friend', 2.2), ('
know', 2.09), ('people', 2.02), ('thing', 2.0), ('day', 1.92), ('rea
lly', 1.83), ('think', 1.51), ('like', 1.51), ('help', 1.48), ('goin
g', 1.38), ('work', 1.37), ('make', 1.33), ('got', 1.3), ('good', 1.
29), ('school', 1.26), ('love', 1.24), ('job', 1.22), ('need', 1.2),
('thought', 1.17), ('way', 1.16), ('better', 1.09), ('self', 1.05),
('month', 1.04), ('say', 1.0), ('lot', 0.98), ('family', 0.95), ('pe
rson', 0.94)]

Topic1 with weights :
[('happy', 8.94), ('make happy', 1.74), ('make', 1.15), ('today', 0.
32), ('finally', 0.3), ('feel happy', 0.27), ('year', 0.25), ('day',
0.24), ('love', 0.23), ('birthday', 0.19), ('happy life', 0.18), ('d
og', 0.18), ('really happy', 0.17), ('happy happy', 0.16), ('want ha
ppy', 0.14), ('smile', 0.14), ('family', 0.13), ('happiness', 0.13),
('happy birthday', 0.12), ('christmas', 0.1), ('picture', 0.1), ('li
ttle', 0.1), ('video', 0.1), ('place', 0.09), ('happy new', 0.09), (
'new', 0.09), ('happy today', 0.09), ('got', 0.09), ('baby', 0.09),
('girl', 0.09)]

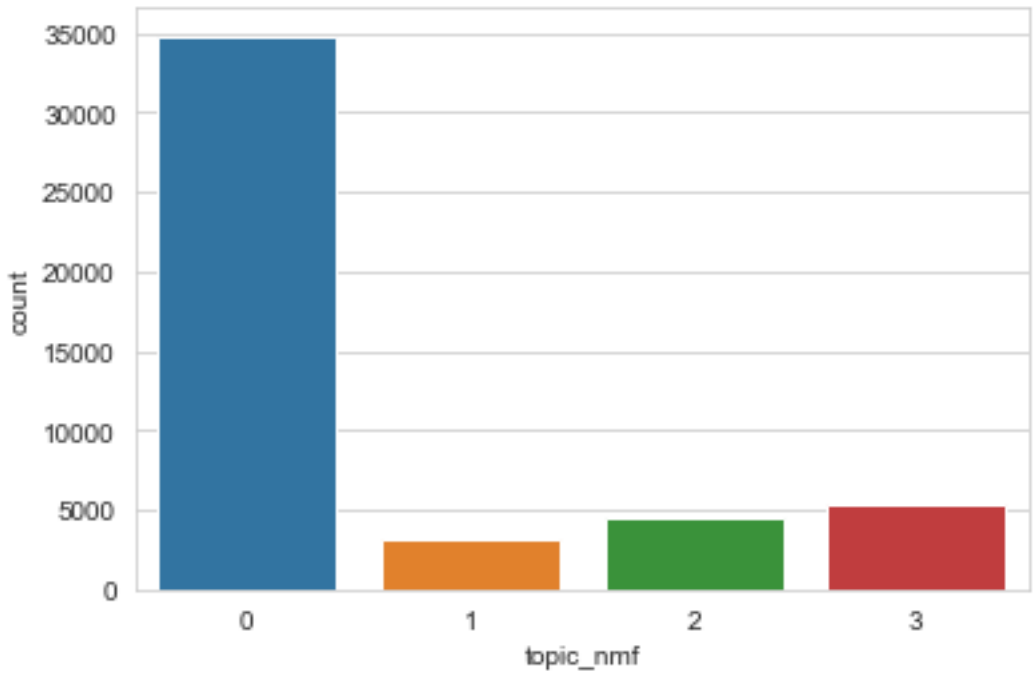
Topic2 with weights :
[('feel', 7.26), ('like', 5.57), ('feel like', 4.96), ('feeling', 0.
83), ('know', 0.78), ('make feel', 0.65), ('really', 0.55), ('make',
0.51), ('people', 0.43), ('dont', 0.41), ('sad', 0.4), ('anymore', 0
.38), ('know feel', 0.32), ('want', 0.31), ('depressed', 0.31), ('li
ke feel', 0.3), ('thing', 0.29), ('like shit', 0.29), ('shit', 0.29)
, ('hate', 0.27), ('want feel', 0.27), ('good', 0.27), ('bad', 0.26)
, ('life feel', 0.26), ('talk', 0.26), ('think', 0.25), ('felt', 0.2
5), ('feeling like', 0.25), ('better', 0.24), ('feel better', 0.23)]

Topic3 with weights :
[('want', 6.51), ('die', 3.54), ('want die', 2.24), ('kill', 2.1), (
'fucking', 1.89), ('anymore', 1.68), ('dont', 1.35), ('hate', 1.13),
('live', 1.07), ('know', 1.02), ('end', 0.89), ('life', 0.88), ('tir
ed', 0.86), ('pain', 0.74), ('want kill', 0.71), ('wish', 0.7), ('ca
re', 0.69), ('dont want', 0.64), ('hurt', 0.62), ('fuck', 0.6), ('wa
nt live', 0.59), ('wanna', 0.56), ('alive', 0.52), ('suicide', 0.52)
, ('want end', 0.52), ('shit', 0.51), ('stop', 0.49), ('death', 0.46

), ('die want', 0.43), ('scared', 0.42)]

/Users/Jay/opt/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:59: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

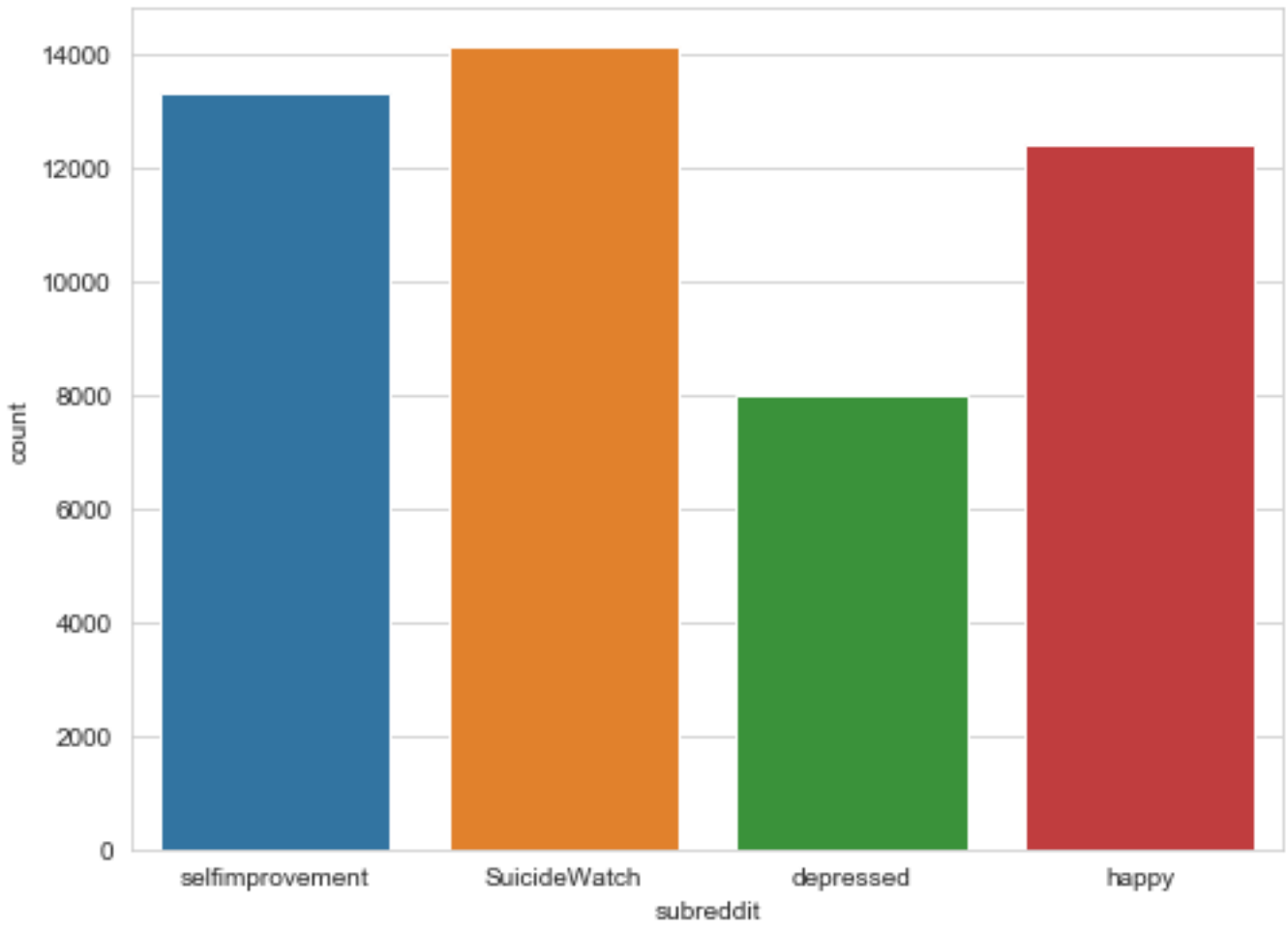


In [25]:

```
# Compare the Bar plot for all 4 subreddit distribution with above Topic modelin
g distribution
plt.figure(figsize=(8,6))
sns.countplot(x='subreddit',data=all_subreddit_df_list)
```

Out[25]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a391c3690>
```



In [26]:

```
from collections import Counter
print(Counter(all_subreddit_df_list.topic_nmf))
```

```
Counter({0: 34804, 3: 5328, 2: 4577, 1: 3119})
```

In [27]:

```
topic0_gr = all_subreddit_df_list[all_subreddit_df_list['topic_nmf'] ==0]
topic1_gr = all_subreddit_df_list[all_subreddit_df_list['topic_nmf'] ==1]
topic2_gr = all_subreddit_df_list[all_subreddit_df_list['topic_nmf'] ==2]
topic3_gr = all_subreddit_df_list[all_subreddit_df_list['topic_nmf'] ==3]
```

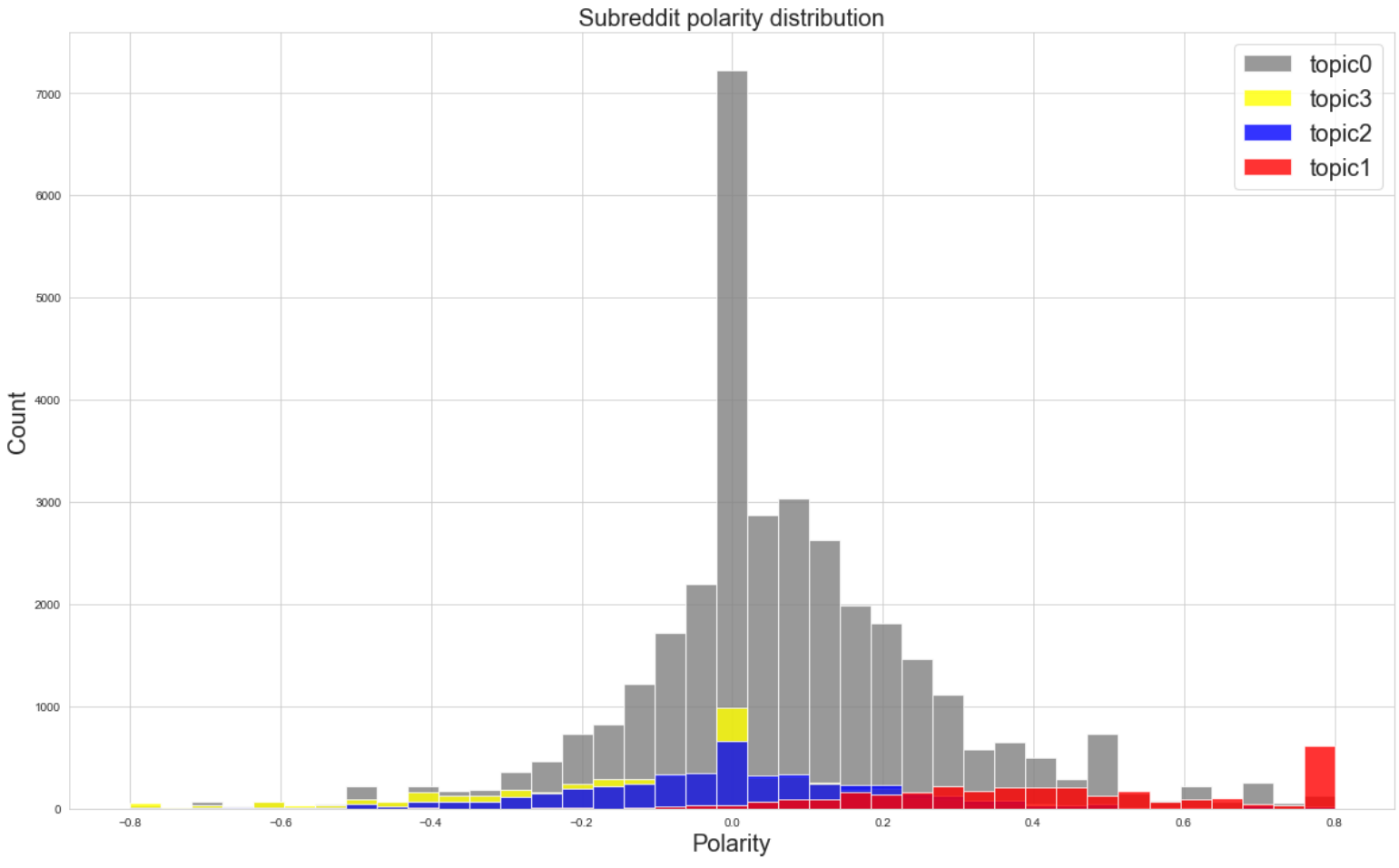
In [28]:

```
from matplotlib import pyplot
plt.figure(figsize=(20,12))
bins = np.linspace(-.8, .8, 40)

pyplot.hist(topic0_gr['polarity'], bins, alpha=0.8, label='topic0',color='grey')
pyplot.hist(topic3_gr['polarity'], bins, alpha=0.8, label='topic3',color='yellow
')
pyplot.hist(topic2_gr['polarity'], bins, alpha=0.8, label='topic2',color='blue')
pyplot.hist(topic1_gr['polarity'], bins, alpha=0.8, label='topic1',color='red')

plt.xlabel('Polarity',fontsize =20)
plt.ylabel('Count',fontsize =20)
pyplot.legend(loc='upper right',fontsize=20)
plt.title('Subreddit polarity distribution',fontsize =20)

pyplot.show()
```



In [29]:

```
# Encoding subreddit into 'subreddit_categorical_label' -> To use this column in
classification modeling
#### 0: SuicideWatch
#### 1 : depressed
#### 2 : happy
#### 3 :  selfimprovement
```

In [30]:

```
from sklearn import preprocessing
le = preprocessing.LabelEncoder()
le.fit(all_subreddit_df_list.subreddit)
all_subreddit_df_list['subreddit_categorical_label'] = le.transform(all_subreddi
t_df_list.subreddit)
```

In [31]:

```
all_subreddit_df_list.head(4)
```

Out[31]:

	author	over_18	title	selftext	num_comments	score	
12970	sudrawkid	False	Can't properly stick up for myself and feel weak.	Hey everyone! I guess I should just outright g...	2	4	https://ww
5192	PinkyIsSnug	False	Gonna kill myself very very soon. I've really ...	I joined this school with high hopes. hopes th...	5	1	https://ww
2357	ReasonableBrother3	False	GET RID OF DEPRESSION	[removed]	0	1	https://ww
35396	Pineapplestick	False	I won my first boxing match!		1	1	https://ww

In [32]:

```
## descriptive statistics after topic modeling
```

In [33]:

```
all_subreddit_df_list.describe(include ='all').transpose()
```

Out[33]:

	count	unique	top	fr
author	47828	30841	[deleted]	20
over_18	47828	2	False	477
title	47828	38485	Help	
selftext	47828	27149		99
num_comments	47828	NaN	NaN	N
score	47828	NaN	NaN	N
full_link	47828	40246	https://www.reddit.com/r/depressed/comments/dq...	
created_utc	47828	NaN	NaN	N
timestamp	47828	40220	2019-11-02 23:41:15	
subreddit	47828	4	SuicideWatch	141
title_with_selftext	47828	40074	I can’t stop crying [removed]	
title_with_selftext_clean	47828	39823		
polarity	47828	NaN	NaN	N
topic_nmf	47828	NaN	NaN	N
subreddit_categorical_label	47828	NaN	NaN	N

In [443]:

```
#Download to csv file for google colab
all_subreddit_df_list.to_csv("all_subreddit_df_list.csv")
```

In [ ]:

```

```

Begin BERT Modeling :

In [252]:

```
import tensorflow as tf
import tensorflow_hub as hub
from datetime import datetime
import bert
from bert import run_classifier
from bert import optimization
from bert import tokenization
from bert import modeling
from tensorflow import keras
import codecs
import tensorflow as tf
from tqdm import tqdm
from chardet import detect
import keras
from keras_radam import RAdam
from keras import backend as K
from keras_bert import load_trained_model_from_checkpoint
from keras_bert import get_base_dict, get_model, compile_model, gen_batch_inputs
import codecs
from keras_bert import Tokenizer
#!pip install keras-bert
#!pip install keras-rectified-adam
```

Train / Test Split

In [270]:

```
seed = 45
X_train, X_test, y_train, y_test = train_test_split(all_subreddit_df_list['title
_with_selftext_clean'], all_subreddit_df_list['subreddit_categorical_label'], \
                                                    test_size=0.33, random_state
=seed)
```

In [271]:

```
X_train.shape, X_test.shape, y_train.shape,y_test.shape,
```

Out[271]:

```
((32044,), (15784,), (32044,), (15784,))
```

In [86]:

```

```

Parameters setting

Given the parmeters setting, the BERT paper recommendation is adopt:

Epochs — range between 2,3,4(recommended) but can explore more for experiment puropose (ex) up to 10 or 20

Batch_size — 4,8,16,32 if we use TPU we can go with 128,256 and so on

Learning rate (For Adam): 5e-5, 3e-5, 2e-5 in paper -> but we will use RAdam with 1e-4 for first trial

Since BERT has already optimized the layers & hidden units for us.

In [146]:

```
SEQ_LEN = 128

BATCH_SIZE = 32

EPOCHS = 4
#Epoch: an arbitrary cutoff, generally defined as "one pass over the entire data set",
# used to separate training into distinct phases, which is useful for logging and periodic evaluation.
# When using validation_data or validation_split with the fit method of Keras models,
# evaluation will be run at the end of every epoch.

LR = 1e-4
```

Build Path to the pretrained model of BERT

In [65]:

```
pretrained_path = 'uncased_L-12_H-768_A-12'
config_path = os.path.join(pretrained_path, 'bert_config.json')
checkpoint_path = os.path.join(pretrained_path, 'bert_model.ckpt')
vocab_path = os.path.join(pretrained_path, 'vocab.txt')
```

Load Pretrained BERT model

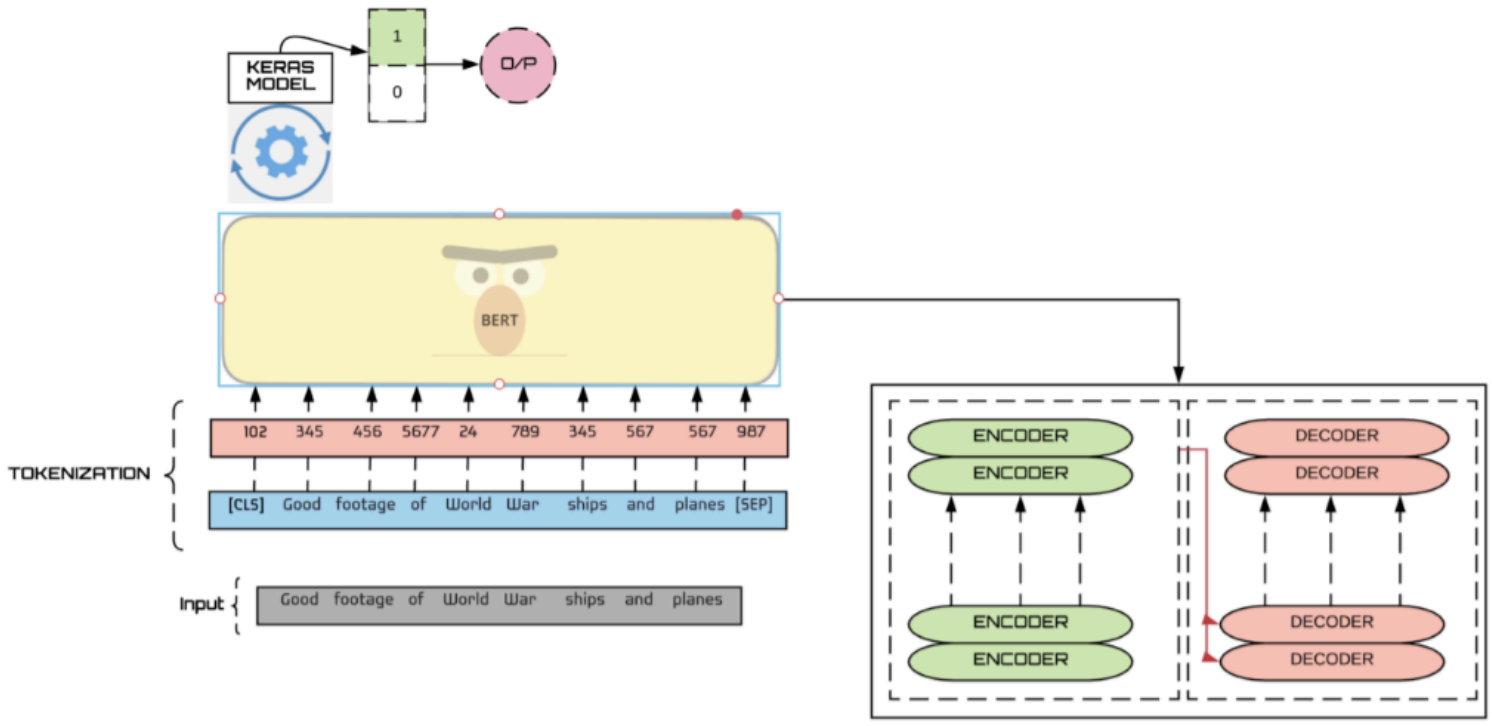
In [77]:

```
model = load_trained_model_from_checkpoint(
      config_path,
      checkpoint_path,
      training=True,
      trainable=True,
      seq_len=SEQ_LEN,
  )
```

Model Architecture and Layers

In [455]:

```
display(Image(filename='bert-architecture.png'))
```



BERT Architecture

The total number of trainable parameters is ~110M, just like the BERT paper mentions. Here's a brief of various steps in the model:

Two inputs: One from word tokens, one from segment-layer

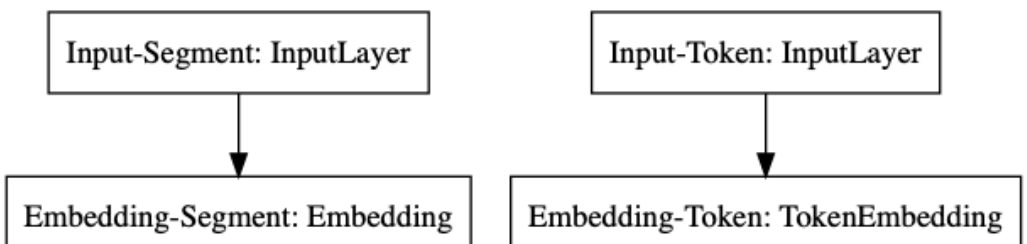
These get added, summed over to a third embedding: position embedding, followed by dropout and a layer normalization

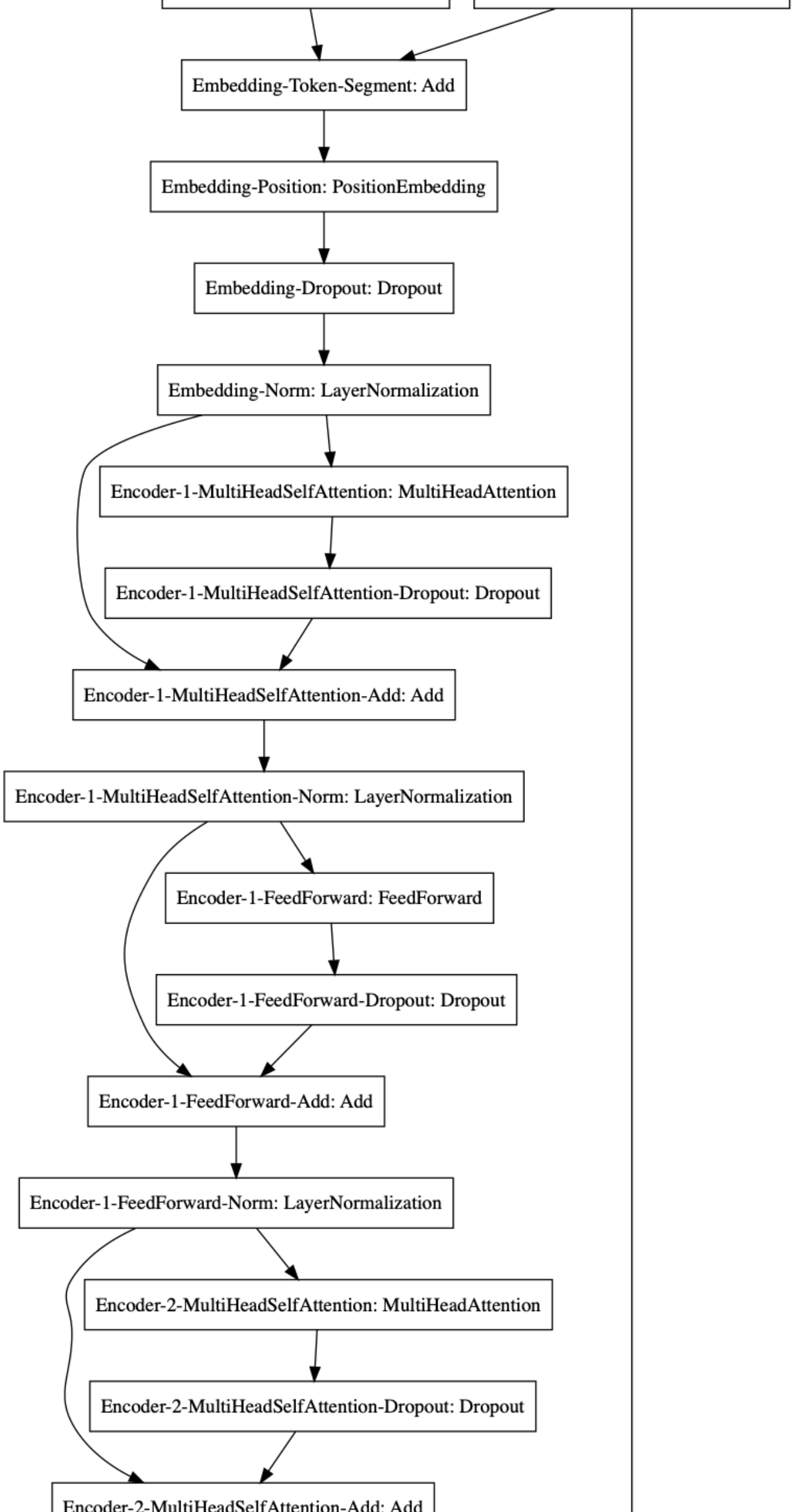
BERT consists of 12 Transformer layers. Each transformer takes in a list of token embeddings, and produces the same number of embeddings on the output (but with the feature values changed).

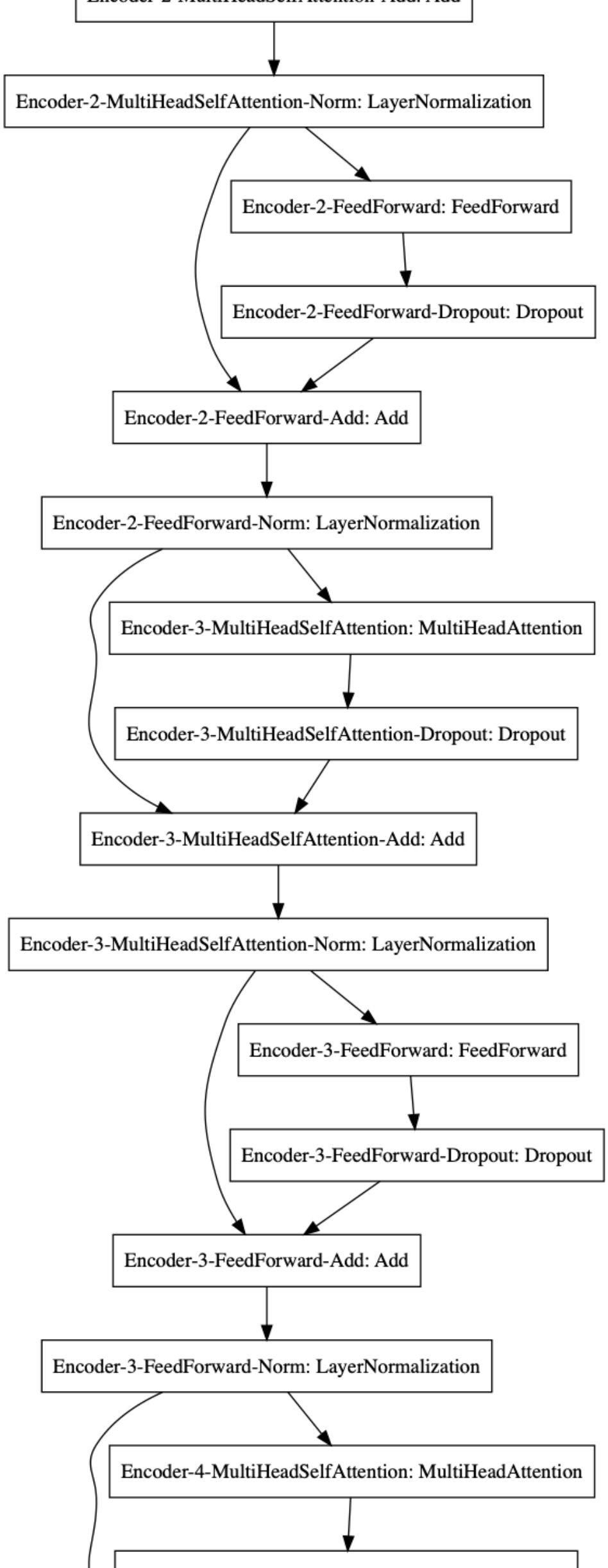
Following these 12 layers, there are two outputs — one for NSP (Next Sentence Prediction) and one for MLM (Masked Language Modeling)

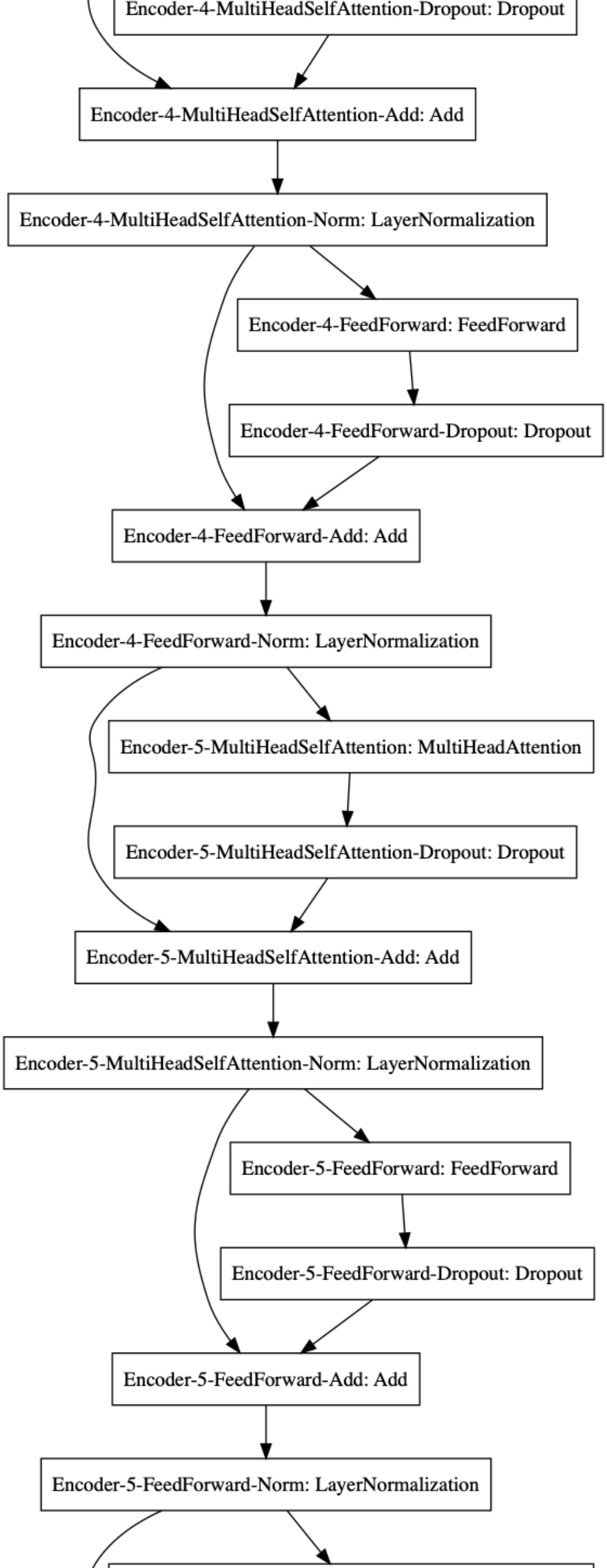
In [437]:

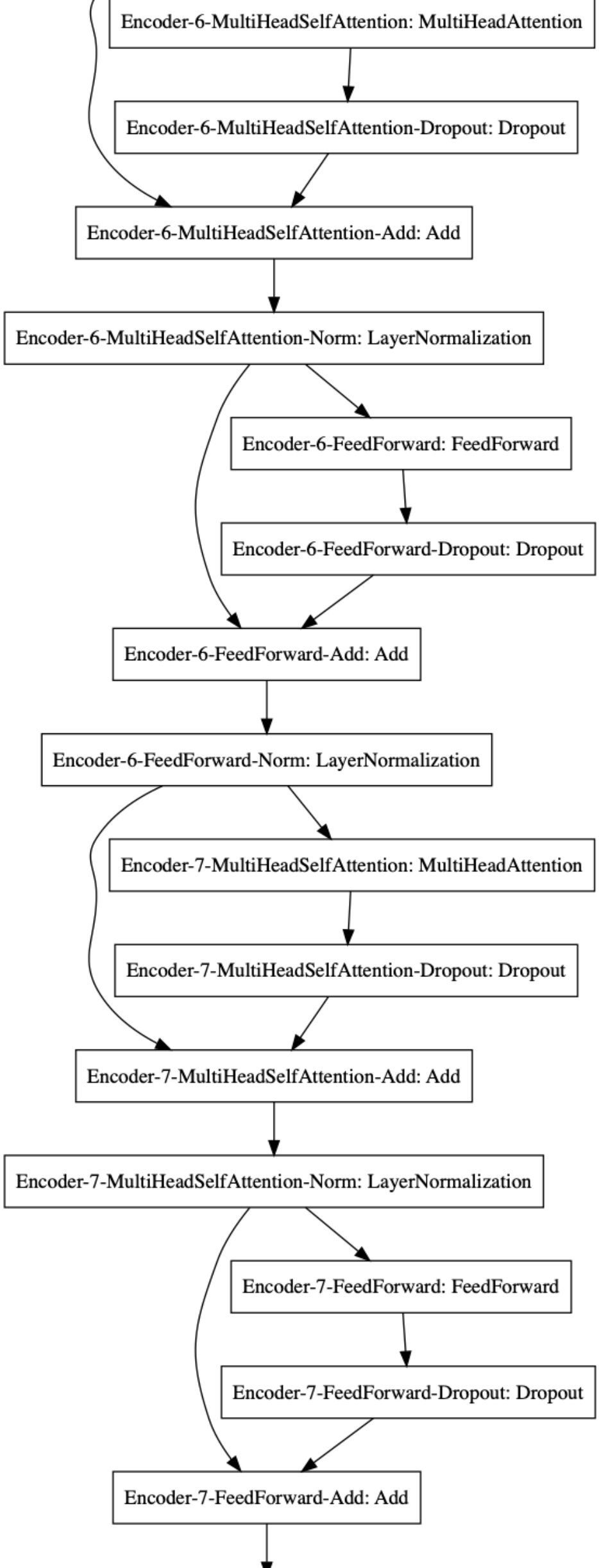
```
display(Image(filename='bert_layers.png'))
```

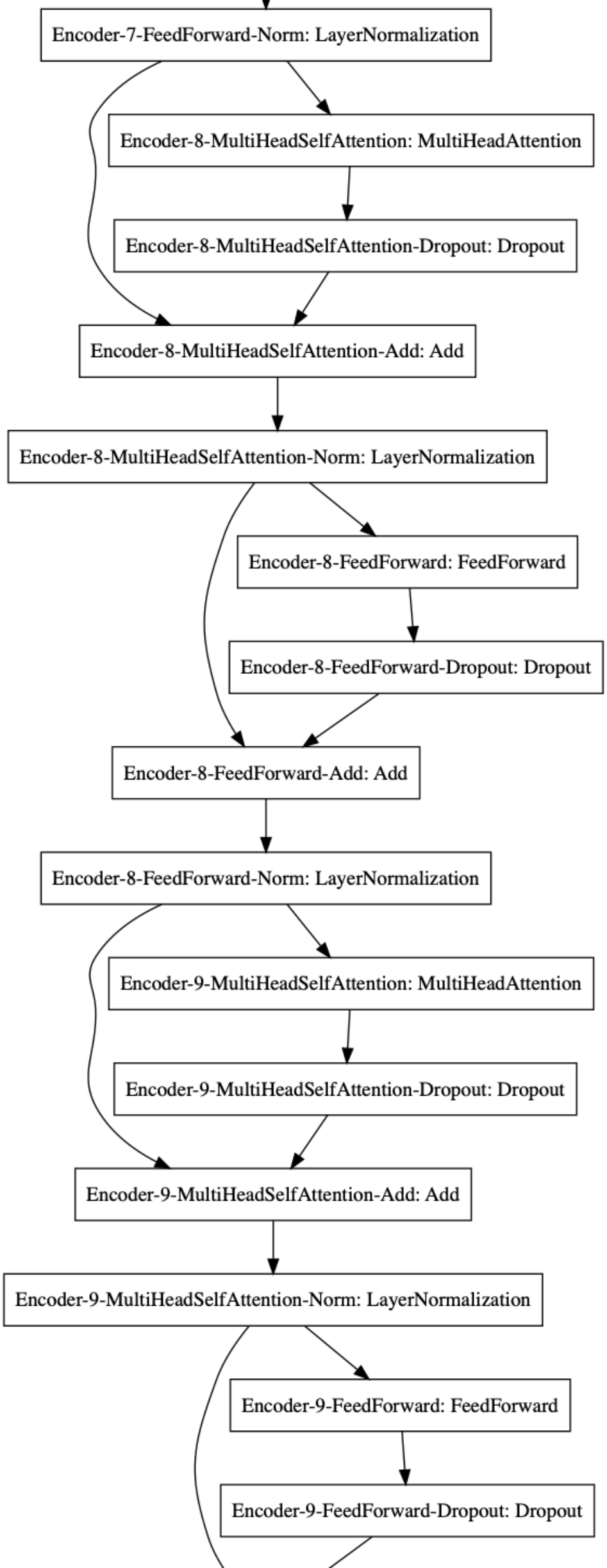


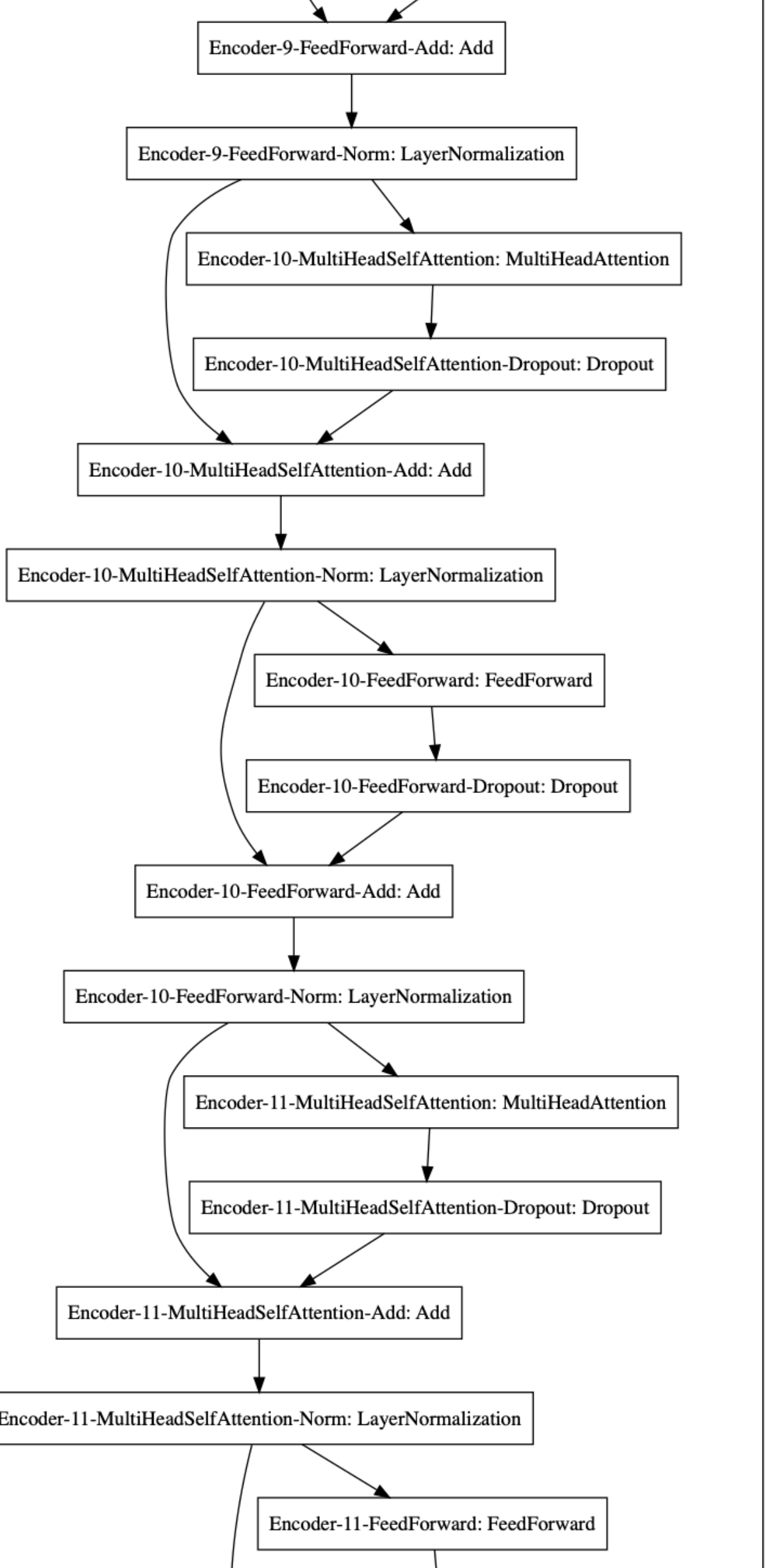

Embedding-Token-Segment: Add
Embedding-Position: PositionEmbedding
Embedding-Dropout: Dropout
Embedding-Norm: LayerNormalization
Encoder-1-MultiHeadSelfAttention: MultiHeadAttention
Encoder-1-MultiHeadSelfAttention-Dropout: Dropout
Encoder-1-MultiHeadSelfAttention-Add: Add
Encoder-1-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-1-FeedForward: FeedForward
Encoder-1-FeedForward-Dropout: Dropout
Encoder-1-FeedForward-Add: Add
Encoder-1-FeedForward-Norm: LayerNormalization
Encoder-2-MultiHeadSelfAttention: MultiHeadAttention
Encoder-2-MultiHeadSelfAttention-Dropout: Dropout
Encoder-2-MultiHeadSelfAttention-Add: Add

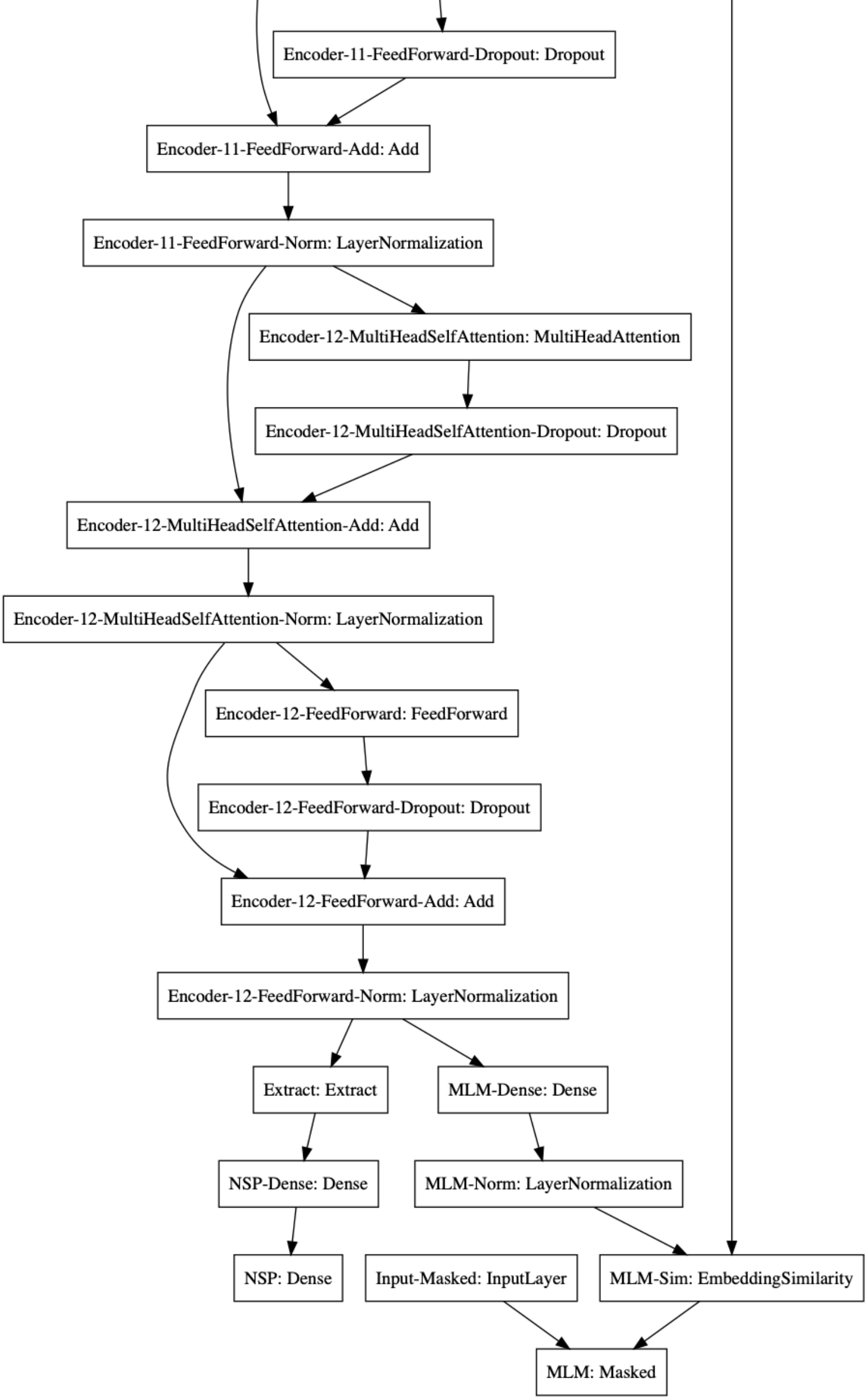

Encoder-2-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-2-FeedForward: FeedForward
Encoder-2-FeedForward-Dropout: Dropout
Encoder-2-FeedForward-Add: Add
Encoder-2-FeedForward-Norm: LayerNormalization
Encoder-3-MultiHeadSelfAttention: MultiHeadAttention
Encoder-3-MultiHeadSelfAttention-Dropout: Dropout
Encoder-3-MultiHeadSelfAttention-Add: Add
Encoder-3-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-3-FeedForward: FeedForward
Encoder-3-FeedForward-Dropout: Dropout
Encoder-3-FeedForward-Add: Add
Encoder-3-FeedForward-Norm: LayerNormalization
Encoder-4-MultiHeadSelfAttention: MultiHeadAttention


Encoder-4-MultiHeadSelfAttention-Dropout: Dropout
Encoder-4-MultiHeadSelfAttention-Add: Add
Encoder-4-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-4-FeedForward: FeedForward
Encoder-4-FeedForward-Dropout: Dropout
Encoder-4-FeedForward-Add: Add
Encoder-4-FeedForward-Norm: LayerNormalization
Encoder-5-MultiHeadSelfAttention: MultiHeadAttention
Encoder-5-MultiHeadSelfAttention-Dropout: Dropout
Encoder-5-MultiHeadSelfAttention-Add: Add
Encoder-5-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-5-FeedForward: FeedForward
Encoder-5-FeedForward-Dropout: Dropout
Encoder-5-FeedForward-Add: Add
Encoder-5-FeedForward-Norm: LayerNormalization


Encoder-6-MultiHeadSelfAttention: MultiHeadAttention
Encoder-6-MultiHeadSelfAttention-Dropout: Dropout
Encoder-6-MultiHeadSelfAttention-Add: Add
Encoder-6-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-6-FeedForward: FeedForward
Encoder-6-FeedForward-Dropout: Dropout
Encoder-6-FeedForward-Add: Add
Encoder-6-FeedForward-Norm: LayerNormalization
Encoder-7-MultiHeadSelfAttention: MultiHeadAttention
Encoder-7-MultiHeadSelfAttention-Dropout: Dropout
Encoder-7-MultiHeadSelfAttention-Add: Add
Encoder-7-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-7-FeedForward: FeedForward
Encoder-7-FeedForward-Dropout: Dropout
Encoder-7-FeedForward-Add: Add


Encoder-7-FeedForward-Norm: LayerNormalization
Encoder-8-MultiHeadSelfAttention: MultiHeadAttention
Encoder-8-MultiHeadSelfAttention-Dropout: Dropout
Encoder-8-MultiHeadSelfAttention-Add: Add
Encoder-8-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-8-FeedForward: FeedForward
Encoder-8-FeedForward-Dropout: Dropout
Encoder-8-FeedForward-Add: Add
Encoder-8-FeedForward-Norm: LayerNormalization
Encoder-9-MultiHeadSelfAttention: MultiHeadAttention
Encoder-9-MultiHeadSelfAttention-Dropout: Dropout
Encoder-9-MultiHeadSelfAttention-Add: Add
Encoder-9-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-9-FeedForward: FeedForward
Encoder-9-FeedForward-Dropout: Dropout


Encoder-9-FeedForward-Add: Add
Encoder-9-FeedForward-Norm: LayerNormalization
Encoder-10-MultiHeadSelfAttention: MultiHeadAttention
Encoder-10-MultiHeadSelfAttention-Dropout: Dropout
Encoder-10-MultiHeadSelfAttention-Add: Add
Encoder-10-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-10-FeedForward: FeedForward
Encoder-10-FeedForward-Dropout: Dropout
Encoder-10-FeedForward-Add: Add
Encoder-10-FeedForward-Norm: LayerNormalization
Encoder-11-MultiHeadSelfAttention: MultiHeadAttention
Encoder-11-MultiHeadSelfAttention-Dropout: Dropout
Encoder-11-MultiHeadSelfAttention-Add: Add
Encoder-11-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-11-FeedForward: FeedForward


Encoder-11-FeedForward-Dropout: Dropout
Encoder-11-FeedForward-Add: Add
Encoder-11-FeedForward-Norm: LayerNormalization
Encoder-12-MultiHeadSelfAttention: MultiHeadAttention
Encoder-12-MultiHeadSelfAttention-Dropout: Dropout
Encoder-12-MultiHeadSelfAttention-Add: Add
Encoder-12-MultiHeadSelfAttention-Norm: LayerNormalization
Encoder-12-FeedForward: FeedForward
Encoder-12-FeedForward-Dropout: Dropout
Encoder-12-FeedForward-Add: Add
Encoder-12-FeedForward-Norm: LayerNormalization
Extract: Extract
MLM-Dense: Dense
NSP-Dense: Dense
MLM-Norm: LayerNormalization
NSP: Dense
Input-Masked: InputLayer
MLM-Sim: EmbeddingSimilarity
MLM: Masked

In [135]:

```
## print model summary  110M trainable params
model.summary()
```

```
__________________________________________________________________________
_____________________________
Layer (type)                    Output Shape         Param #     Con
nected to
==========================================================================
=============================
Input-Token (InputLayer)        (None, 128)          0
__________________________________________________________________________
_____________________________
Input-Segment (InputLayer)      (None, 128)          0
__________________________________________________________________________
_____________________________
Embedding-Token (TokenEmbedding [(None, 128, 768), ( 23440896    Inp
ut-Token[0][0]
__________________________________________________________________________
_____________________________
Embedding-Segment (Embedding)   (None, 128, 768)     1536        Inp
ut-Segment[0][0]
__________________________________________________________________________
_____________________________
Embedding-Token-Segment (Add)   (None, 128, 768)     0           Emb
edding-Token[0][0]

Embedding-Segment[0][0]
__________________________________________________________________________
_____________________________
Embedding-Position (PositionEmb (None, 128, 768)     98304       Emb
edding-Token-Segment[0][0]
__________________________________________________________________________
_____________________________
Embedding-Dropout (Dropout)     (None, 128, 768)     0           Emb
edding-Position[0][0]
__________________________________________________________________________
_____________________________
Embedding-Norm (LayerNormalizat (None, 128, 768)     1536        Emb
edding-Dropout[0][0]
__________________________________________________________________________
_____________________________
Encoder-1-MultiHeadSelfAttentio (None, 128, 768)     2362368     Emb
edding-Norm[0][0]
__________________________________________________________________________
_____________________________
Encoder-1-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-1-MultiHeadSelfAttention[
__________________________________________________________________________
_____________________________
Encoder-1-MultiHeadSelfAttentio (None, 128, 768)     0           Emb
edding-Norm[0][0]
```

Encoder-1-MultiHeadSelfAttention-

Encoder-1-MultiHeadSelfAttentio (None, 128, 768) 1536 Enc
oder-1-MultiHeadSelfAttention-

Encoder-1-FeedForward (FeedForw (None, 128, 768) 4722432 Enc
oder-1-MultiHeadSelfAttention-

Encoder-1-FeedForward-Dropout ((None, 128, 768) 0 Enc
oder-1-FeedForward[0][0]

Encoder-1-FeedForward-Add (Add) (None, 128, 768) 0 Enc
oder-1-MultiHeadSelfAttention-

Encoder-1-FeedForward-Dropout[0][

Encoder-1-FeedForward-Norm (Lay (None, 128, 768) 1536 Enc
oder-1-FeedForward-Add[0][0]

Encoder-2-MultiHeadSelfAttentio (None, 128, 768) 2362368 Enc
oder-1-FeedForward-Norm[0][0]

Encoder-2-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-2-MultiHeadSelfAttention[

Encoder-2-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-1-FeedForward-Norm[0][0]

Encoder-2-MultiHeadSelfAttention-

Encoder-2-MultiHeadSelfAttentio (None, 128, 768) 1536 Enc
oder-2-MultiHeadSelfAttention-

Encoder-2-FeedForward (FeedForw (None, 128, 768) 4722432 Enc
oder-2-MultiHeadSelfAttention-

Encoder-2-FeedForward-Dropout ((None, 128, 768) 0 Enc
oder-2-FeedForward[0][0]

Encoder-2-FeedForward-Add (Add) (None, 128, 768) 0 Enc

oder-2-MultiHeadSelfAttention-

Encoder-2-FeedForward-Dropout[0][

Encoder-2-FeedForward-Norm (Lay (None, 128, 768) 1536 Enc
oder-2-FeedForward-Add[0][0]

Encoder-3-MultiHeadSelfAttentio (None, 128, 768) 2362368 Enc
oder-2-FeedForward-Norm[0][0]

Encoder-3-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-3-MultiHeadSelfAttention[

Encoder-3-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-2-FeedForward-Norm[0][0]

Encoder-3-MultiHeadSelfAttention-

Encoder-3-MultiHeadSelfAttentio (None, 128, 768) 1536 Enc
oder-3-MultiHeadSelfAttention-

Encoder-3-FeedForward (FeedForw (None, 128, 768) 4722432 Enc
oder-3-MultiHeadSelfAttention-

Encoder-3-FeedForward-Dropout ((None, 128, 768) 0 Enc
oder-3-FeedForward[0][0]

Encoder-3-FeedForward-Add (Add) (None, 128, 768) 0 Enc
oder-3-MultiHeadSelfAttention-

Encoder-3-FeedForward-Dropout[0][

Encoder-3-FeedForward-Norm (Lay (None, 128, 768) 1536 Enc
oder-3-FeedForward-Add[0][0]

Encoder-4-MultiHeadSelfAttentio (None, 128, 768) 2362368 Enc
oder-3-FeedForward-Norm[0][0]

Encoder-4-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-4-MultiHeadSelfAttention[

Encoder-4-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-3-FeedForward-Norm[0][0]

Encoder-4-MultiHeadSelfAttention-

Encoder-4-MultiHeadSelfAttentio (None, 128, 768) 1536 Enc
oder-4-MultiHeadSelfAttention-

Encoder-4-FeedForward (FeedForw (None, 128, 768) 4722432 Enc
oder-4-MultiHeadSelfAttention-

Encoder-4-FeedForward-Dropout ((None, 128, 768) 0 Enc
oder-4-FeedForward[0][0]

Encoder-4-FeedForward-Add (Add) (None, 128, 768) 0 Enc
oder-4-MultiHeadSelfAttention-

Encoder-4-FeedForward-Dropout[0][

Encoder-4-FeedForward-Norm (Lay (None, 128, 768) 1536 Enc
oder-4-FeedForward-Add[0][0]

Encoder-5-MultiHeadSelfAttentio (None, 128, 768) 2362368 Enc
oder-4-FeedForward-Norm[0][0]

Encoder-5-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-5-MultiHeadSelfAttention[

Encoder-5-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-4-FeedForward-Norm[0][0]

Encoder-5-MultiHeadSelfAttention-

Encoder-5-MultiHeadSelfAttentio (None, 128, 768) 1536 Enc
oder-5-MultiHeadSelfAttention-

Encoder-5-FeedForward (FeedForw (None, 128, 768) 4722432 Enc
oder-5-MultiHeadSelfAttention-

Encoder-5-FeedForward-Dropout ((None, 128, 768) 0 Enc
oder-5-FeedForward[0][0]

```
_____________________________________
Encoder-5-FeedForward-Add (Add) (None, 128, 768)     0           Enc
oder-5-MultiHeadSelfAttention-

Encoder-5-FeedForward-Dropout[0][
_____________________________________________________________________
_____________________________________
Encoder-5-FeedForward-Norm (Lay (None, 128, 768)     1536        Enc
oder-5-FeedForward-Add[0][0]
_____________________________________________________________________
_____________________________________
Encoder-6-MultiHeadSelfAttentio (None, 128, 768)     2362368     Enc
oder-5-FeedForward-Norm[0][0]
_____________________________________________________________________
_____________________________________
Encoder-6-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-6-MultiHeadSelfAttention[
_____________________________________________________________________
_____________________________________
Encoder-6-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-5-FeedForward-Norm[0][0]

Encoder-6-MultiHeadSelfAttention-
_____________________________________________________________________
_____________________________________
Encoder-6-MultiHeadSelfAttentio (None, 128, 768)     1536        Enc
oder-6-MultiHeadSelfAttention-
_____________________________________________________________________
_____________________________________
Encoder-6-FeedForward (FeedForw (None, 128, 768)     4722432     Enc
oder-6-MultiHeadSelfAttention-
_____________________________________________________________________
_____________________________________
Encoder-6-FeedForward-Dropout ( (None, 128, 768)     0           Enc
oder-6-FeedForward[0][0]
_____________________________________________________________________
_____________________________________
Encoder-6-FeedForward-Add (Add) (None, 128, 768)     0           Enc
oder-6-MultiHeadSelfAttention-

Encoder-6-FeedForward-Dropout[0][
_____________________________________________________________________
_____________________________________
Encoder-6-FeedForward-Norm (Lay (None, 128, 768)     1536        Enc
oder-6-FeedForward-Add[0][0]
_____________________________________________________________________
_____________________________________
Encoder-7-MultiHeadSelfAttentio (None, 128, 768)     2362368     Enc
oder-6-FeedForward-Norm[0][0]
_____________________________________________________________________
_____________________________________
Encoder-7-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-7-MultiHeadSelfAttention[
```

```
_____
_____
Encoder-7-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-6-FeedForward-Norm[0][0]

Encoder-7-MultiHeadSelfAttention-
_____
_____
Encoder-7-MultiHeadSelfAttentio (None, 128, 768)     1536        Enc
oder-7-MultiHeadSelfAttention-
_____
_____
Encoder-7-FeedForward (FeedForw (None, 128, 768)     4722432     Enc
oder-7-MultiHeadSelfAttention-
_____
_____
Encoder-7-FeedForward-Dropout ( (None, 128, 768)     0           Enc
oder-7-FeedForward[0][0]
_____
_____
Encoder-7-FeedForward-Add (Add) (None, 128, 768)     0           Enc
oder-7-MultiHeadSelfAttention-

Encoder-7-FeedForward-Dropout[0][
_____
_____
Encoder-7-FeedForward-Norm (Lay (None, 128, 768)     1536        Enc
oder-7-FeedForward-Add[0][0]
_____
_____
Encoder-8-MultiHeadSelfAttentio (None, 128, 768)     2362368     Enc
oder-7-FeedForward-Norm[0][0]
_____
_____
Encoder-8-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-8-MultiHeadSelfAttention[
_____
_____
Encoder-8-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-7-FeedForward-Norm[0][0]

Encoder-8-MultiHeadSelfAttention-
_____
_____
Encoder-8-MultiHeadSelfAttentio (None, 128, 768)     1536        Enc
oder-8-MultiHeadSelfAttention-
_____
_____
Encoder-8-FeedForward (FeedForw (None, 128, 768)     4722432     Enc
oder-8-MultiHeadSelfAttention-
_____
_____
Encoder-8-FeedForward-Dropout ( (None, 128, 768)     0           Enc
```

oder-8-FeedForward[0][0]

Encoder-8-FeedForward-Add (Add) (None, 128, 768) 0 Enc
oder-8-MultiHeadSelfAttention-

Encoder-8-FeedForward-Dropout[0][

Encoder-8-FeedForward-Norm (Lay (None, 128, 768) 1536 Enc
oder-8-FeedForward-Add[0][0]

Encoder-9-MultiHeadSelfAttentio (None, 128, 768) 2362368 Enc
oder-8-FeedForward-Norm[0][0]

Encoder-9-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-9-MultiHeadSelfAttention[

Encoder-9-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-8-FeedForward-Norm[0][0]

Encoder-9-MultiHeadSelfAttention-

Encoder-9-MultiHeadSelfAttentio (None, 128, 768) 1536 Enc
oder-9-MultiHeadSelfAttention-

Encoder-9-FeedForward (FeedForw (None, 128, 768) 4722432 Enc
oder-9-MultiHeadSelfAttention-

Encoder-9-FeedForward-Dropout ((None, 128, 768) 0 Enc
oder-9-FeedForward[0][0]

Encoder-9-FeedForward-Add (Add) (None, 128, 768) 0 Enc
oder-9-MultiHeadSelfAttention-

Encoder-9-FeedForward-Dropout[0][

Encoder-9-FeedForward-Norm (Lay (None, 128, 768) 1536 Enc
oder-9-FeedForward-Add[0][0]

Encoder-10-MultiHeadSelfAttenti (None, 128, 768) 2362368 Enc
oder-9-FeedForward-Norm[0][0]

Encoder-10-MultiHeadSelfAttenti oder-10-MultiHeadSelfAttention	(None, 128, 768)	0	Enc
Encoder-10-MultiHeadSelfAttenti oder-9-FeedForward-Norm[0][0] Encoder-10-MultiHeadSelfAttention	(None, 128, 768)	0	Enc
Encoder-10-MultiHeadSelfAttenti oder-10-MultiHeadSelfAttention	(None, 128, 768)	1536	Enc
Encoder-10-FeedForward (FeedFor oder-10-MultiHeadSelfAttention	(None, 128, 768)	4722432	Enc
Encoder-10-FeedForward-Dropout oder-10-FeedForward[0][0]	(None, 128, 768)	0	Enc
Encoder-10-FeedForward-Add (Add oder-10-MultiHeadSelfAttention Encoder-10-FeedForward-Dropout[0]	(None, 128, 768)	0	Enc
Encoder-10-FeedForward-Norm (La oder-10-FeedForward-Add[0][0]	(None, 128, 768)	1536	Enc
Encoder-11-MultiHeadSelfAttenti oder-10-FeedForward-Norm[0][0]	(None, 128, 768)	2362368	Enc
Encoder-11-MultiHeadSelfAttenti oder-11-MultiHeadSelfAttention	(None, 128, 768)	0	Enc
Encoder-11-MultiHeadSelfAttenti oder-10-FeedForward-Norm[0][0] Encoder-11-MultiHeadSelfAttention	(None, 128, 768)	0	Enc
Encoder-11-MultiHeadSelfAttenti oder-11-MultiHeadSelfAttention	(None, 128, 768)	1536	Enc
Encoder-11-FeedForward (FeedFor oder-11-MultiHeadSelfAttention	(None, 128, 768)	4722432	Enc

Encoder-11-FeedForward-Dropout	(None, 128, 768)	0	Encoder-11-FeedForward[0][0]
Encoder-11-FeedForward-Add (Add	(None, 128, 768)	0	Encoder-11-MultiHeadSelfAttention Encoder-11-FeedForward-Dropout[0]
Encoder-11-FeedForward-Norm (La	(None, 128, 768)	1536	Encoder-11-FeedForward-Add[0][0]
Encoder-12-MultiHeadSelfAttenti	(None, 128, 768)	2362368	Encoder-11-FeedForward-Norm[0][0]
Encoder-12-MultiHeadSelfAttenti	(None, 128, 768)	0	Encoder-12-MultiHeadSelfAttention
Encoder-12-MultiHeadSelfAttenti	(None, 128, 768)	0	Encoder-11-FeedForward-Norm[0][0] Encoder-12-MultiHeadSelfAttention
Encoder-12-MultiHeadSelfAttenti	(None, 128, 768)	1536	Encoder-12-MultiHeadSelfAttention
Encoder-12-FeedForward (FeedFor	(None, 128, 768)	4722432	Encoder-12-MultiHeadSelfAttention
Encoder-12-FeedForward-Dropout	(None, 128, 768)	0	Encoder-12-FeedForward[0][0]
Encoder-12-FeedForward-Add (Add	(None, 128, 768)	0	Encoder-12-MultiHeadSelfAttention Encoder-12-FeedForward-Dropout[0]
Encoder-12-FeedForward-Norm (La	(None, 128, 768)	1536	Encoder-12-FeedForward-Add[0][0]
MLM-Dense (Dense)	(None, 128, 768)	590592	Encoder-12-FeedForward-Norm[0][0]

```
__________________________________________________________________________________
_____________________________________
MLM-Norm (LayerNormalization)   (None, 128, 768)     1536        MLM
-Dense[0][0]
__________________________________________________________________________________
_____________________________________
Extract (Extract)               (None, 768)          0           Enc
oder-12-FeedForward-Norm[0][0]
__________________________________________________________________________________
_____________________________________
MLM-Sim (EmbeddingSimilarity)   (None, 128, 30522)   30522       MLM
-Norm[0][0]

Embedding-Token[0][1]
__________________________________________________________________________________
_____________________________________
Input-Masked (InputLayer)       (None, 128)          0
__________________________________________________________________________________
_____________________________________
NSP-Dense (Dense)               (None, 768)          590592      Ext
ract[0][0]
__________________________________________________________________________________
_____________________________________
MLM (Masked)                    (None, 128, 30522)   0           MLM
-Sim[0][0]

Input-Masked[0][0]
__________________________________________________________________________________
_____________________________________
NSP (Dense)                     (None, 2)            1538        NSP
-Dense[0][0]
==================================================================================
==============================
Total params: 109,811,516
Trainable params: 109,811,516
Non-trainable params: 0
__________________________________________________________________________________
_____________________________________
```

In [ ]:

```
# Extracting token dictionary from vocab of pretrained model to refer
```

In [483]:

```
token_dict = {}
with codecs.open(vocab_path, 'r', 'utf8') as reader:
    for line in reader:
        token = line.strip()
        token_dict[token] = len(token_dict)
```

Initiating tokenizer

“[CLS]” and “[SEP]” tokens at the beginning and at the end of each sequence.

As BERT model requests, token “[CLS]” stands for class and has to be placed at the beginning of the input example. “[SEP]” token is for separating sentences for the next sentence prediction task.

In [81]:

```
tokenizer = Tokenizer(token_dict)
```

In [82]:

```
print(tokenizer.tokenize('unaffable'))
```

```
['[CLS]', 'una', '##ffa', '##ble', '[SEP]']
```

In [83]:

```
indices, segments = tokenizer.encode('unaffable')
print(indices)
print(segments)
```

```
[101, 14477, 20961, 3468, 102]
[0, 0, 0, 0, 0]
```

In [139]:

```
model.inputs
```

Out[139]:

```
[<tf.Tensor 'Input-Token_1:0' shape=(?, 128) dtype=float32>,
 <tf.Tensor 'Input-Segment_1:0' shape=(?, 128) dtype=float32>,
 <tf.Tensor 'Input-Masked_1:0' shape=(?, 128) dtype=float32>]
```

In [ ]:

```
#X_train, X_test, y_train, y_test
```

In [272]:

```
X_train.values
```

Out[272]:

```
array([' 20m  continuosly feeling like need signifcant  tip feeling
whole single  hey   year old man  ever since relationship  longest r
elationship first girlfriend dated  afterwards  grieved week quickly
decided go partying drinking single thing able  however  quickly hit
excessive partying missed companion  specifically  individual rock
started talking girl quickly  finding new relationship month later
quickly realized ready  instead plagued mind thought girl right one
know realize also ready relationship  week later  talking new girl
recent ex  started friend benefit quickly turned relationship  long
story short  broke due personal family problem    year later  relat
ionship  finding wanting break cycle  however  continuously find fee
ling lonely  friend  job  go school  continue feel void partner  you
ng  know  reason  always feel like much better seeing someone  anyon
e tip grow person feel reliance girlfriend      thank ',
       'time year ago unemployed  hated expect live another year  ph
oto taken yesterday  new house  wonderful girlfriend two year two ne
w kitten  life really get better  ',
       'bad birthday  hand  trying make sound like greedy person  3
0th september become  received least one present card parent friend
happened year row  honesty  work butt please parent good grade rang
ing  spending endless night studying room avoiding distraction  eve
n teacher awed work  however  feel like everyone forget birthday ev
en worse parent  least one person came today said  happy birthday
think cause  lone wolf school socially insecure viewed  akward weird
hardly pester someone annoying request bully  usually help people sc
hool work  telling  always one person buy birthday present someone
finally  buy cake sing happy birthday isolated room  know wrong
also first reddit post ',
       ..., 'everybody able pull together get nephew switch  happy
',
       'best friend year old happy girl  ',
       'almost day since employed  many interview turned  today offe
red job love city love paid ever paid  smile abound freaking happy
felt hopeless languishing parent house  finally get move back move l
ife  '],
      dtype=object)
```

Transforming function to convert the Training set into model's input

In [275]:

```
indices= []
for example in X_train.values[:500]:
    ids,segments =tokenizer.encode(example, max_len=SEQ_LEN)
    indices.append(ids)
```

In [277]:

```
np.array(indices)
```

Out[277]:

```
array([[ 101, 2322, 2213, ..., 2514, 2066,  102],
       [ 101, 2051, 2095, ...,    0,    0,    0],
       [ 101, 2919, 5798, ...,    0,    0,    0],
       ...,
       [ 101, 2667, 6865, ...,    0,    0,    0],
       [ 101, 2342, 3191, ...,    0,    0,    0],
       [ 101, 9906, 2371, ...,    0,    0,    0]])
```

In [278]:

```
np.array(indices).shape[0] ,np.array(indices).shape[1]
```

Out[278]:

```
(500, 128)
```

In [279]:

```
[np.array(indices),np.zeros_like(np.array(indices))] # for input-token, segment-token
```

Out[279]:

```
[array([[ 101, 2322, 2213, ..., 2514, 2066,  102],
        [ 101, 2051, 2095, ...,    0,    0,    0],
        [ 101, 2919, 5798, ...,    0,    0,    0],
        ...,
        [ 101, 2667, 6865, ...,    0,    0,    0],
        [ 101, 2342, 3191, ...,    0,    0,    0],
        [ 101, 9906, 2371, ...,    0,    0,    0]]),
 array([[0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        ...,
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0],
        [0, 0, 0, ..., 0, 0, 0]])]
```

In [280]:

```
X_train =[np.array(indices),np.zeros_like(np.array(indices))]
```

In [282]:

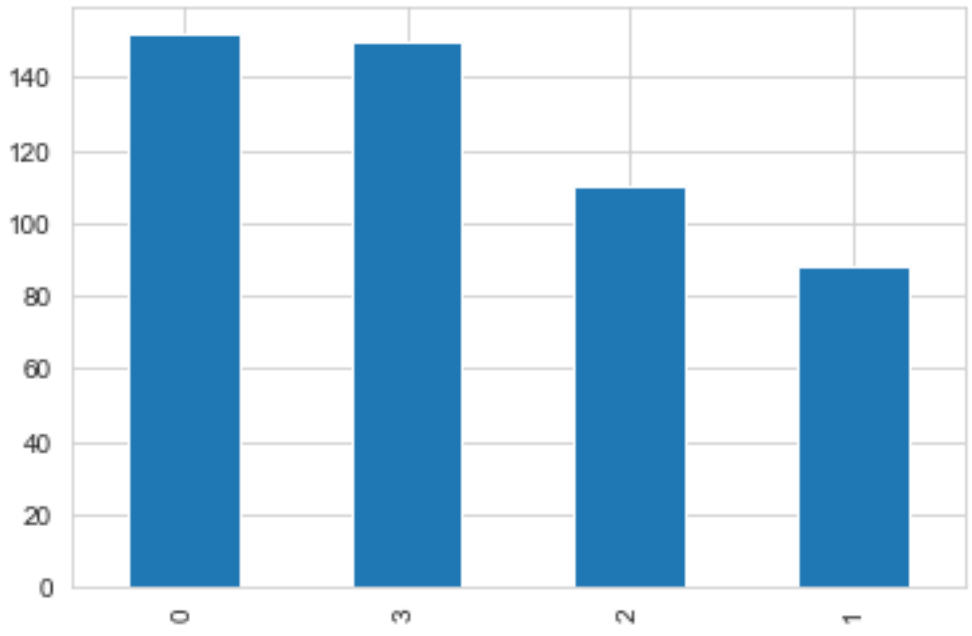
```
y_train = y_train.values[:500]
```

In [283]:

```
pd.Series(y_train).value_counts().plot(kind = 'bar')
```

Out[283]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1b178982d0>
```



In [284]:

```
indices= []
for example in X_test.values[:50]:
    ids,segments =tokenizer.encode(example, max_len=SEQ_LEN)
    indices.append(ids)
```

In [285]:

```
X_test = [np.array(indices),np.zeros_like(np.array(indices))]
```

In [287]:

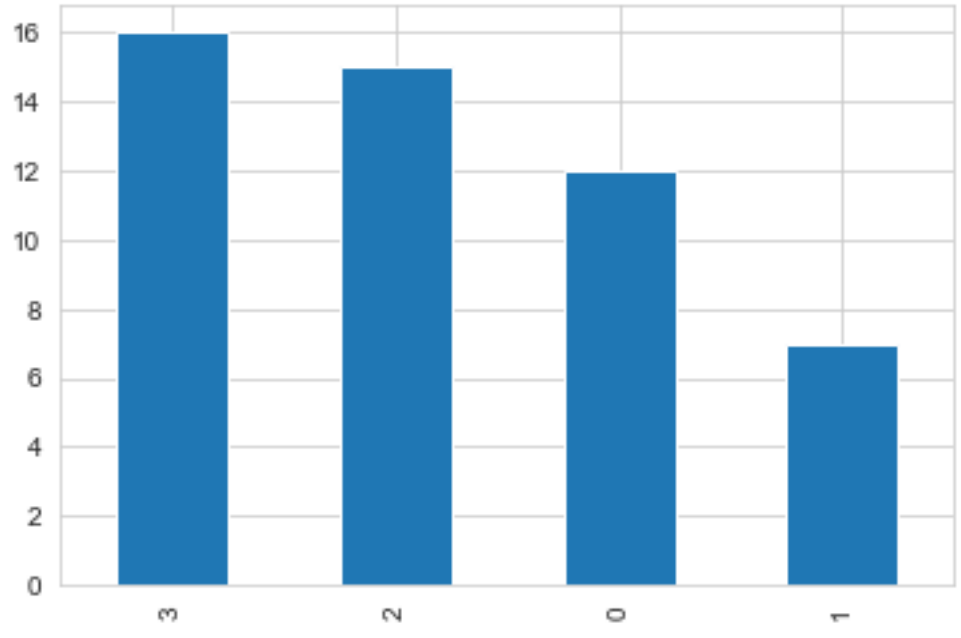
```
y_test = y_test.values[:50]
```

In [288]:

```
pd.Series(y_test).value_counts().plot(kind = 'bar')
```

Out[288]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1ecf3fd0d0>
```



In [ ]:

Fine-tuning

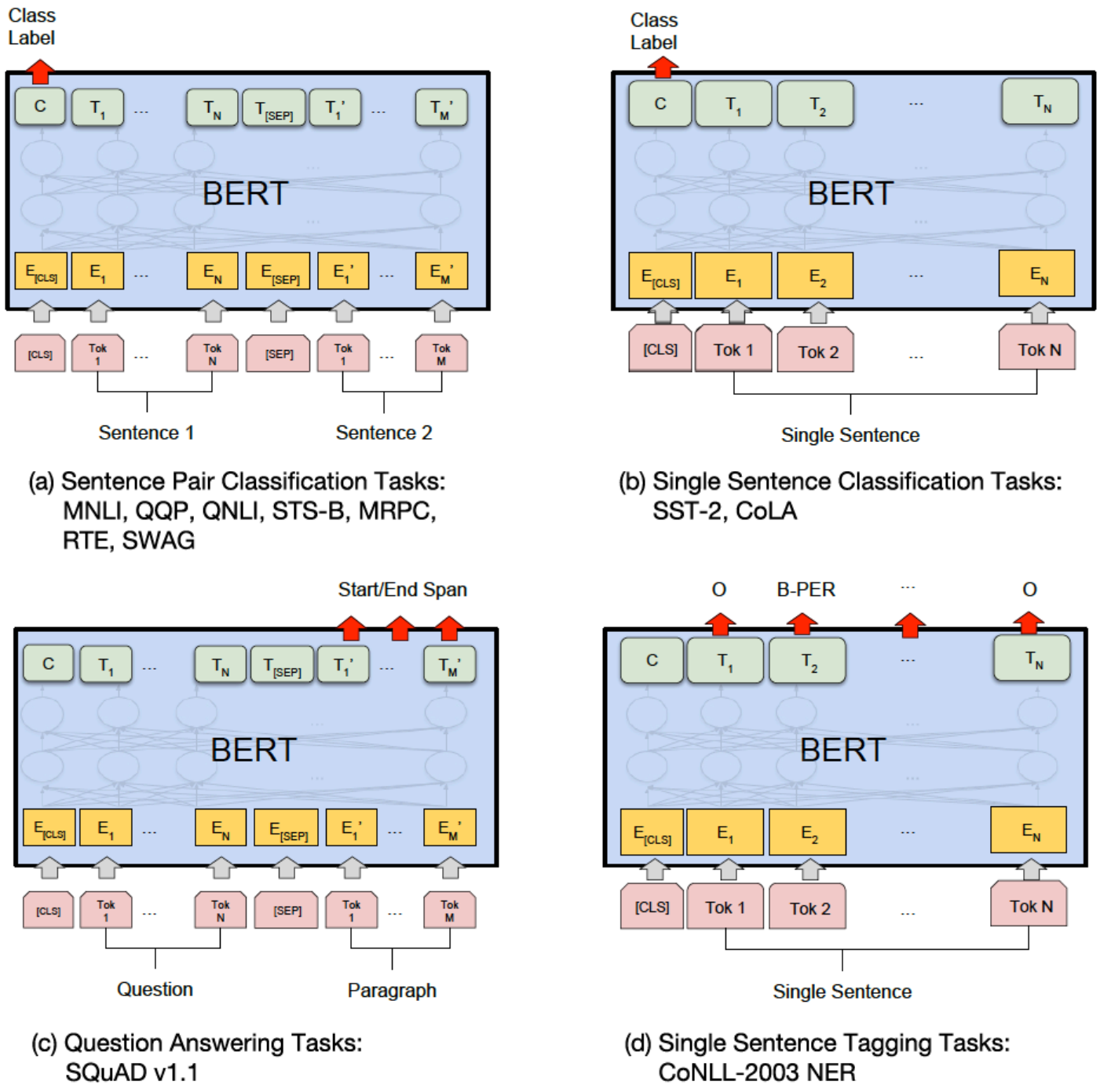
Unlike recent language representation models , BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers.

As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task- specific architecture modifications.

Our case? Extracting layer from pretrained bert model and adding a layer with softmax for 4 classes of subreddit

In [458]:

```
display(Image(filename='fine-tuning-bert.png'))
```



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Figure 4: Illustrations of Fine-tuning BERT on Different Tasks.

In [247]:

```
#input_segments represent the separation.
model.inputs[:2]
```

Out[247]:

```
[<tf.Tensor 'Input-Token_1:0' shape=(?, 128) dtype=float32>,
 <tf.Tensor 'Input-Segment_1:0' shape=(?, 128) dtype=float32>]
```

In [ ]:

```
# Assign inputs and output layers for the BERT model
# there are two outputs — one for NSP (Next Sentence Prediction) and one for MLM
(Masked Language Modeling)
# For classification NSP-Dense layer will be needed

inputs = model.inputs[:2]
dense = model.get_layer('NSP-Dense').output
#NSP-Dense is the first dense layer after the output of [CLS] token.
outputs = keras.layers.Dense(units=4, activation='softmax')(dense)

'''BERT is designed to pretrain deep bidirectional representations from unlabele
d text by jointly conditioning on both
left and right context in all layers. As a result, the pre-trained BERT model ca
n be finetuned with just one additional output layer
to create state-of-the-art models for a wide range of tasks, such as question an
swering and language inference,
without  substantial taskspecific architecture modifications.
(2019) BERT: Pre-training of Deep Bidirectional Transformers for Language Unders
tanding'''
```

In [ ]:

```
#Model Build/Compile
model = keras.models.Model(inputs, outputs)

model.compile(
  RAdam(learning_rate = LR), # We can add up warmup_proportion ex) 0.1

  loss='sparse_categorical_crossentropy',

'''For sparse_categorical_crossentropy, For class 1 and class 2 targets, in a 5-
class classification problem,
the list should be [1,2].  Basically, the targets should be in integer form in o
rder to call sparse_categorical_crossentropy.
This is called sparse since the target representation requires much less space t
han one-hot encoding.
For example, a batch with b targets and k classes needs b * k space to be repres
ented in one-hot,
whereas a batch with b targets and k classes needs b space to be represented in
integer form.

For categorical_crossentropy, for class 1 and class 2 targets, in a 5-class clas
sification problem,
the list should be [[0,1,0,0,0], [0,0,1,0,0]].
Basically, the targets should be in one-hot form in order to call categorical_cr
ossentropy
representation of the targets are the only difference,
the results should be the same since they are both calculating categorical cross
entropy.
'''
  metrics=['sparse_categorical_accuracy'],

 # keras will choose the maximum value from this array and check if it correspon
ds to the index of the max value
# in y_pred, should only provide an integer of the true class
# Categorical Accuracy: It evaluates the index of the maximal true value is equa
l to the index of the maximal predicted
#value. you need to specify your target (y) as one-hot encoded vector
)
```

Optimizer RAdam and why :

Rectified Adam optimizer:

A good optimizer trains models fast, but it also prevents them from getting stuck in a local minimum. Rectified Adam is a one of the most recent deep learning model optimizers introduced by a collaboration between members of the University of Illinois, Georgia Tech, and Microsoft Research.

Essentially, they seek to understand why a "warmup" phase is beneficial for scheduling learning rates, and then "identify" the underlying problem to be related to" high variance" and "poor generalization" during the first few batches.

They find that the issue can be remedied by using either a "warmup" or "low initial learning rate",or by turning off momentum for the first couple of batches. nstead of setting the learning rate alpha as a constant or in a decreasing order, a learning rate warmup strategy sets αt as smaller values in the first few steps, and as more training examples are fed in, the variance stabilizes and the learning rate/momentum can be increased. They therefore proposed a Rectified Adam optimizer that dynamically changes the momentum in a way that hedges against high variance.

The goal of the Rectified Adam optimizer is :

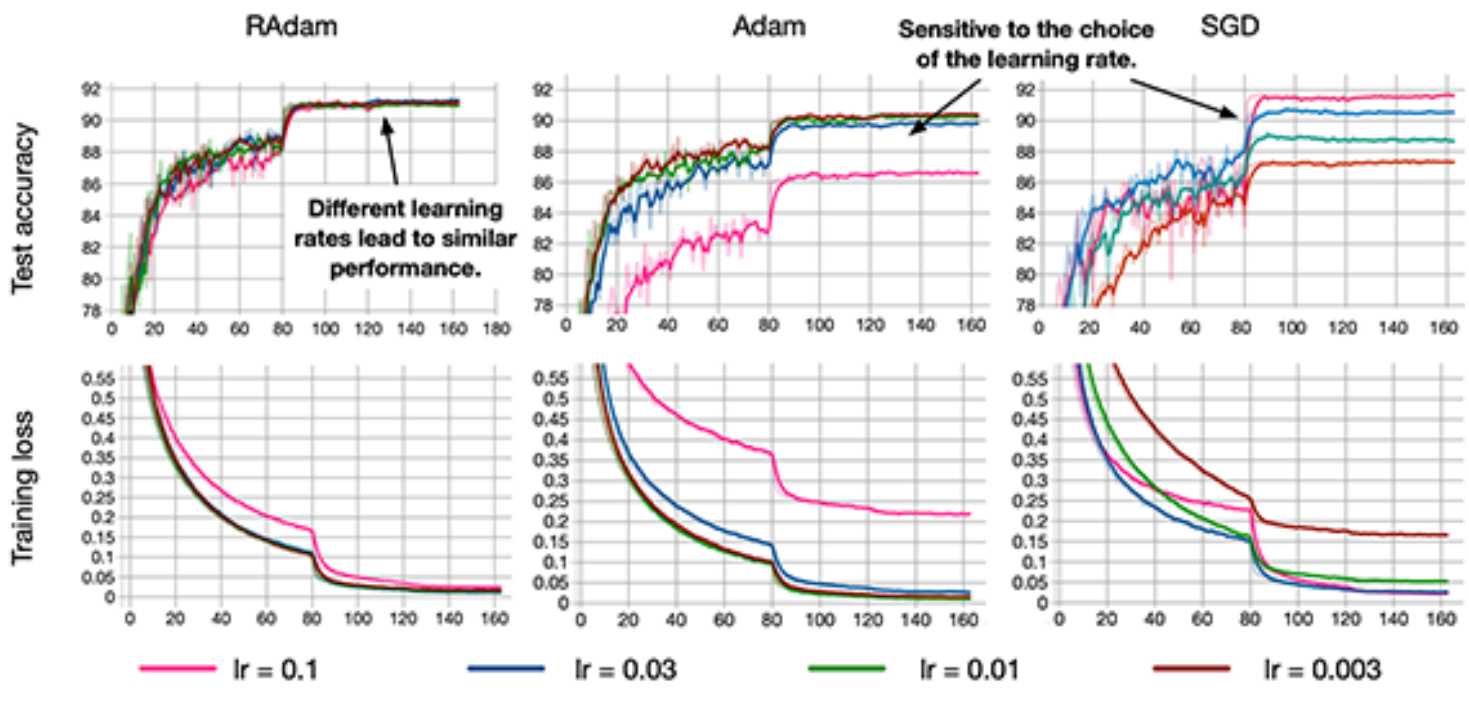
1) Obtain a more robust/more generalizable deep neural network

2) dynamically changes the learning rate/momentum & hedges against high variance.

3) To solve the problem of the adaptive learning rate (i.e.,problematically large variance in the early stage), suggest warmup works as a variance reduction technique, and provide both empirical and theoretical evidence to verify our hypothesis

In [436]:

```
display(Image(filename='rectified_adam.png'))
```



In [482]:

```
# Initializing variables.

## Variable tensors are used when the values require updating within a session.
### It is the type of tensor that would be used for the weights matrix when crea
ting neural networks,
### since these values will be updated as the model is being trained.
```

In [215]:

```
sess = K.get_session()
uninitialized_variables = set([i.decode('ascii') for i in sess.run(tf.report_uni
nitialized_variables())])
init_op = tf.variables_initializer(
    [v for v in tf.global_variables() if v.name.split(':')[0] in uninitialized_v
ariables]
)
sess.run(init_op)
```

Begin Training and Validation for 4 epcohs

In [290]:

```
history= model.fit(
    X_train,
    y_train,
    validation_split = 0.15,
    shuffle =True,
    epochs=4,
    batch_size=BATCH_SIZE,
)
```

```
Train on 425 samples, validate on 75 samples
Epoch 1/4
425/425 [==============================] - 2590s 6s/step - loss: 0.7
831 - sparse_categorical_accuracy: 0.6729 - val_loss: 0.9129 - val_s
parse_categorical_accuracy: 0.6933
Epoch 2/4
425/425 [==============================] - 2569s 6s/step - loss: 0.4
713 - sparse_categorical_accuracy: 0.8447 - val_loss: 0.9049 - val_s
parse_categorical_accuracy: 0.6667
Epoch 3/4
425/425 [==============================] - 2421s 6s/step - loss: 0.2
546 - sparse_categorical_accuracy: 0.9271 - val_loss: 1.0616 - val_s
parse_categorical_accuracy: 0.6667
Epoch 4/4
425/425 [==============================] - 2306s 5s/step - loss: 0.1
345 - sparse_categorical_accuracy: 0.9694 - val_loss: 1.0292 - val_s
parse_categorical_accuracy: 0.6800
```

In [331]:

```
# list all data in history
history.history
```

Out[331]:

```
{'val_loss': [0.9128828414281209,
  0.9048531874020894,
  1.0615886640548706,
  1.0292242685953776],
 'val_sparse_categorical_accuracy': [0.6933333357175191,
  0.6666666650772095,
  0.6666666690508525,
  0.6799999984105428],
 'loss': [0.783128489746767,
  0.4712752474055571,
  0.2546408712162691,
  0.13452121325275476],
 'sparse_categorical_accuracy': [0.6729411768913269,
  0.8447058824931875,
  0.927058823669658,
  0.9694117647058823]}
```

In [434]:

```
model.summary()
```

```
__________________________________________________________________________________
_____________________________
Layer (type)                    Output Shape         Param #     Con
nected to
==================================================================================
=============================
Input-Token (InputLayer)        (None, 128)          0
__________________________________________________________________________________
_____________________________
Input-Segment (InputLayer)      (None, 128)          0
__________________________________________________________________________________
_____________________________
Embedding-Token (TokenEmbedding [(None, 128, 768), ( 23440896    Inp
ut-Token[0][0]
__________________________________________________________________________________
_____________________________
Embedding-Segment (Embedding)   (None, 128, 768)     1536        Inp
ut-Segment[0][0]
__________________________________________________________________________________
_____________________________
Embedding-Token-Segment (Add)   (None, 128, 768)     0           Emb
edding-Token[0][0]

Embedding-Segment[0][0]
__________________________________________________________________________________
_____________________________
```

Embedding-Position (PositionEmb (None, 128, 768) 98304 Emb
edding-Token-Segment[0][0]

Embedding-Dropout (Dropout) (None, 128, 768) 0 Emb
edding-Position[0][0]

Embedding-Norm (LayerNormalizat (None, 128, 768) 1536 Emb
edding-Dropout[0][0]

Encoder-1-MultiHeadSelfAttentio (None, 128, 768) 2362368 Emb
edding-Norm[0][0]

Encoder-1-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-1-MultiHeadSelfAttention[

Encoder-1-MultiHeadSelfAttentio (None, 128, 768) 0 Emb
edding-Norm[0][0]

Encoder-1-MultiHeadSelfAttention-

Encoder-1-MultiHeadSelfAttentio (None, 128, 768) 1536 Enc
oder-1-MultiHeadSelfAttention-

Encoder-1-FeedForward (FeedForw (None, 128, 768) 4722432 Enc
oder-1-MultiHeadSelfAttention-

Encoder-1-FeedForward-Dropout ((None, 128, 768) 0 Enc
oder-1-FeedForward[0][0]

Encoder-1-FeedForward-Add (Add) (None, 128, 768) 0 Enc
oder-1-MultiHeadSelfAttention-

Encoder-1-FeedForward-Dropout[0][

Encoder-1-FeedForward-Norm (Lay (None, 128, 768) 1536 Enc
oder-1-FeedForward-Add[0][0]

Encoder-2-MultiHeadSelfAttentio (None, 128, 768) 2362368 Enc
oder-1-FeedForward-Norm[0][0]

Encoder-2-MultiHeadSelfAttentio (None, 128, 768) 0 Enc

oder-2-MultiHeadSelfAttention[

Encoder-2-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-1-FeedForward-Norm[0][0]

Encoder-2-MultiHeadSelfAttention-

Encoder-2-MultiHeadSelfAttentio (None, 128, 768) 1536 Enc
oder-2-MultiHeadSelfAttention-

Encoder-2-FeedForward (FeedForw (None, 128, 768) 4722432 Enc
oder-2-MultiHeadSelfAttention-

Encoder-2-FeedForward-Dropout ((None, 128, 768) 0 Enc
oder-2-FeedForward[0][0]

Encoder-2-FeedForward-Add (Add) (None, 128, 768) 0 Enc
oder-2-MultiHeadSelfAttention-

Encoder-2-FeedForward-Dropout[0][

Encoder-2-FeedForward-Norm (Lay (None, 128, 768) 1536 Enc
oder-2-FeedForward-Add[0][0]

Encoder-3-MultiHeadSelfAttentio (None, 128, 768) 2362368 Enc
oder-2-FeedForward-Norm[0][0]

Encoder-3-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-3-MultiHeadSelfAttention[

Encoder-3-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-2-FeedForward-Norm[0][0]

Encoder-3-MultiHeadSelfAttention-

Encoder-3-MultiHeadSelfAttentio (None, 128, 768) 1536 Enc
oder-3-MultiHeadSelfAttention-

Encoder-3-FeedForward (FeedForw (None, 128, 768) 4722432 Enc
oder-3-MultiHeadSelfAttention-

```
Encoder-3-FeedForward-Dropout ( (None, 128, 768)     0           Enc
oder-3-FeedForward[0][0]
_________________________________________________________________
_________________________________
Encoder-3-FeedForward-Add (Add) (None, 128, 768)     0           Enc
oder-3-MultiHeadSelfAttention-

Encoder-3-FeedForward-Dropout[0][
_________________________________________________________________
_________________________________
Encoder-3-FeedForward-Norm (Lay (None, 128, 768)     1536        Enc
oder-3-FeedForward-Add[0][0]
_________________________________________________________________
_________________________________
Encoder-4-MultiHeadSelfAttentio (None, 128, 768)     2362368     Enc
oder-3-FeedForward-Norm[0][0]
_________________________________________________________________
_________________________________
Encoder-4-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-4-MultiHeadSelfAttention[
_________________________________________________________________
_________________________________
Encoder-4-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-3-FeedForward-Norm[0][0]

Encoder-4-MultiHeadSelfAttention-
_________________________________________________________________
_________________________________
Encoder-4-MultiHeadSelfAttentio (None, 128, 768)     1536        Enc
oder-4-MultiHeadSelfAttention-
_________________________________________________________________
_________________________________
Encoder-4-FeedForward (FeedForw (None, 128, 768)     4722432     Enc
oder-4-MultiHeadSelfAttention-
_________________________________________________________________
_________________________________
Encoder-4-FeedForward-Dropout ( (None, 128, 768)     0           Enc
oder-4-FeedForward[0][0]
_________________________________________________________________
_________________________________
Encoder-4-FeedForward-Add (Add) (None, 128, 768)     0           Enc
oder-4-MultiHeadSelfAttention-

Encoder-4-FeedForward-Dropout[0][
_________________________________________________________________
_________________________________
Encoder-4-FeedForward-Norm (Lay (None, 128, 768)     1536        Enc
oder-4-FeedForward-Add[0][0]
_________________________________________________________________
_________________________________
Encoder-5-MultiHeadSelfAttentio (None, 128, 768)     2362368     Enc
oder-4-FeedForward-Norm[0][0]
_________________________________________________________________
```

```
_____________________________
Encoder-5-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-5-MultiHeadSelfAttention[
_____________________________________________________________________

_____________________________
Encoder-5-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-4-FeedForward-Norm[0][0]

Encoder-5-MultiHeadSelfAttention-
_____________________________________________________________________

_____________________________
Encoder-5-MultiHeadSelfAttentio (None, 128, 768)     1536        Enc
oder-5-MultiHeadSelfAttention-
_____________________________________________________________________

_____________________________
Encoder-5-FeedForward (FeedForw (None, 128, 768)     4722432     Enc
oder-5-MultiHeadSelfAttention-
_____________________________________________________________________

_____________________________
Encoder-5-FeedForward-Dropout ( (None, 128, 768)     0           Enc
oder-5-FeedForward[0][0]
_____________________________________________________________________

_____________________________
Encoder-5-FeedForward-Add (Add) (None, 128, 768)     0           Enc
oder-5-MultiHeadSelfAttention-

Encoder-5-FeedForward-Dropout[0][
_____________________________________________________________________

_____________________________
Encoder-5-FeedForward-Norm (Lay (None, 128, 768)     1536        Enc
oder-5-FeedForward-Add[0][0]
_____________________________________________________________________

_____________________________
Encoder-6-MultiHeadSelfAttentio (None, 128, 768)     2362368     Enc
oder-5-FeedForward-Norm[0][0]
_____________________________________________________________________

_____________________________
Encoder-6-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-6-MultiHeadSelfAttention[
_____________________________________________________________________

_____________________________
Encoder-6-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-5-FeedForward-Norm[0][0]

Encoder-6-MultiHeadSelfAttention-
_____________________________________________________________________

_____________________________
Encoder-6-MultiHeadSelfAttentio (None, 128, 768)     1536        Enc
oder-6-MultiHeadSelfAttention-
_____________________________________________________________________

_____________________________
Encoder-6-FeedForward (FeedForw (None, 128, 768)     4722432     Enc
oder-6-MultiHeadSelfAttention-
```

```
_____________________________________
Encoder-6-FeedForward-Dropout ( (None, 128, 768)     0           Enc
oder-6-FeedForward[0][0]
_____________________________________
_____________________________________
Encoder-6-FeedForward-Add (Add) (None, 128, 768)     0           Enc
oder-6-MultiHeadSelfAttention-

Encoder-6-FeedForward-Dropout[0][
_____________________________________
_____________________________________
Encoder-6-FeedForward-Norm (Lay (None, 128, 768)     1536        Enc
oder-6-FeedForward-Add[0][0]
_____________________________________
_____________________________________
Encoder-7-MultiHeadSelfAttentio (None, 128, 768)     2362368     Enc
oder-6-FeedForward-Norm[0][0]
_____________________________________
_____________________________________
Encoder-7-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-7-MultiHeadSelfAttention[
_____________________________________
_____________________________________
Encoder-7-MultiHeadSelfAttentio (None, 128, 768)     0           Enc
oder-6-FeedForward-Norm[0][0]
_____________________________________
Encoder-7-MultiHeadSelfAttention-
_____________________________________
_____________________________________
Encoder-7-MultiHeadSelfAttentio (None, 128, 768)     1536        Enc
oder-7-MultiHeadSelfAttention-
_____________________________________
_____________________________________
Encoder-7-FeedForward (FeedForw (None, 128, 768)     4722432     Enc
oder-7-MultiHeadSelfAttention-
_____________________________________
_____________________________________
Encoder-7-FeedForward-Dropout ( (None, 128, 768)     0           Enc
oder-7-FeedForward[0][0]
_____________________________________
_____________________________________
Encoder-7-FeedForward-Add (Add) (None, 128, 768)     0           Enc
oder-7-MultiHeadSelfAttention-

Encoder-7-FeedForward-Dropout[0][
_____________________________________
_____________________________________
Encoder-7-FeedForward-Norm (Lay (None, 128, 768)     1536        Enc
oder-7-FeedForward-Add[0][0]
_____________________________________
_____________________________________
Encoder-8-MultiHeadSelfAttentio (None, 128, 768)     2362368     Enc
```

oder-7-FeedForward-Norm[0][0]

Encoder-8-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-8-MultiHeadSelfAttention[

Encoder-8-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-7-FeedForward-Norm[0][0]

Encoder-8-MultiHeadSelfAttention-

Encoder-8-MultiHeadSelfAttentio (None, 128, 768) 1536 Enc
oder-8-MultiHeadSelfAttention-

Encoder-8-FeedForward (FeedForw (None, 128, 768) 4722432 Enc
oder-8-MultiHeadSelfAttention-

Encoder-8-FeedForward-Dropout ((None, 128, 768) 0 Enc
oder-8-FeedForward[0][0]

Encoder-8-FeedForward-Add (Add) (None, 128, 768) 0 Enc
oder-8-MultiHeadSelfAttention-

Encoder-8-FeedForward-Dropout[0][

Encoder-8-FeedForward-Norm (Lay (None, 128, 768) 1536 Enc
oder-8-FeedForward-Add[0][0]

Encoder-9-MultiHeadSelfAttentio (None, 128, 768) 2362368 Enc
oder-8-FeedForward-Norm[0][0]

Encoder-9-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-9-MultiHeadSelfAttention[

Encoder-9-MultiHeadSelfAttentio (None, 128, 768) 0 Enc
oder-8-FeedForward-Norm[0][0]

Encoder-9-MultiHeadSelfAttention-

Encoder-9-MultiHeadSelfAttentio (None, 128, 768) 1536 Enc
oder-9-MultiHeadSelfAttention-

Encoder-9-FeedForward (FeedForw oder-9-MultiHeadSelfAttention-	(None, 128, 768)	4722432	Enc
Encoder-9-FeedForward-Dropout (oder-9-FeedForward[0][0]	(None, 128, 768)	0	Enc
Encoder-9-FeedForward-Add (Add) oder-9-MultiHeadSelfAttention-	(None, 128, 768)	0	Enc
Encoder-9-FeedForward-Dropout[0][			
Encoder-9-FeedForward-Norm (Lay oder-9-FeedForward-Add[0][0]	(None, 128, 768)	1536	Enc
Encoder-10-MultiHeadSelfAttenti oder-9-FeedForward-Norm[0][0]	(None, 128, 768)	2362368	Enc
Encoder-10-MultiHeadSelfAttenti oder-10-MultiHeadSelfAttention	(None, 128, 768)	0	Enc
Encoder-10-MultiHeadSelfAttenti oder-9-FeedForward-Norm[0][0]	(None, 128, 768)	0	Enc
Encoder-10-MultiHeadSelfAttention			
Encoder-10-MultiHeadSelfAttenti oder-10-MultiHeadSelfAttention	(None, 128, 768)	1536	Enc
Encoder-10-FeedForward (FeedFor oder-10-MultiHeadSelfAttention	(None, 128, 768)	4722432	Enc
Encoder-10-FeedForward-Dropout oder-10-FeedForward[0][0]	(None, 128, 768)	0	Enc
Encoder-10-FeedForward-Add (Add oder-10-MultiHeadSelfAttention	(None, 128, 768)	0	Enc
Encoder-10-FeedForward-Dropout[0]			
Encoder-10-FeedForward-Norm (La oder-10-FeedForward-Add[0][0]	(None, 128, 768)	1536	Enc

Encoder-11-MultiHeadSelfAttenti oder-10-FeedForward-Norm[0][0]	(None, 128, 768)	2362368	Enc
Encoder-11-MultiHeadSelfAttenti oder-11-MultiHeadSelfAttention	(None, 128, 768)	0	Enc
Encoder-11-MultiHeadSelfAttenti oder-10-FeedForward-Norm[0][0] Encoder-11-MultiHeadSelfAttention	(None, 128, 768)	0	Enc
Encoder-11-MultiHeadSelfAttenti oder-11-MultiHeadSelfAttention	(None, 128, 768)	1536	Enc
Encoder-11-FeedForward (FeedFor oder-11-MultiHeadSelfAttention	(None, 128, 768)	4722432	Enc
Encoder-11-FeedForward-Dropout oder-11-FeedForward[0][0]	(None, 128, 768)	0	Enc
Encoder-11-FeedForward-Add (Add oder-11-MultiHeadSelfAttention Encoder-11-FeedForward-Dropout[0]	(None, 128, 768)	0	Enc
Encoder-11-FeedForward-Norm (La oder-11-FeedForward-Add[0][0]	(None, 128, 768)	1536	Enc
Encoder-12-MultiHeadSelfAttenti oder-11-FeedForward-Norm[0][0]	(None, 128, 768)	2362368	Enc
Encoder-12-MultiHeadSelfAttenti oder-12-MultiHeadSelfAttention	(None, 128, 768)	0	Enc
Encoder-12-MultiHeadSelfAttenti oder-11-FeedForward-Norm[0][0] Encoder-12-MultiHeadSelfAttention	(None, 128, 768)	0	Enc
Encoder-12-MultiHeadSelfAttenti oder-12-MultiHeadSelfAttention	(None, 128, 768)	1536	Enc

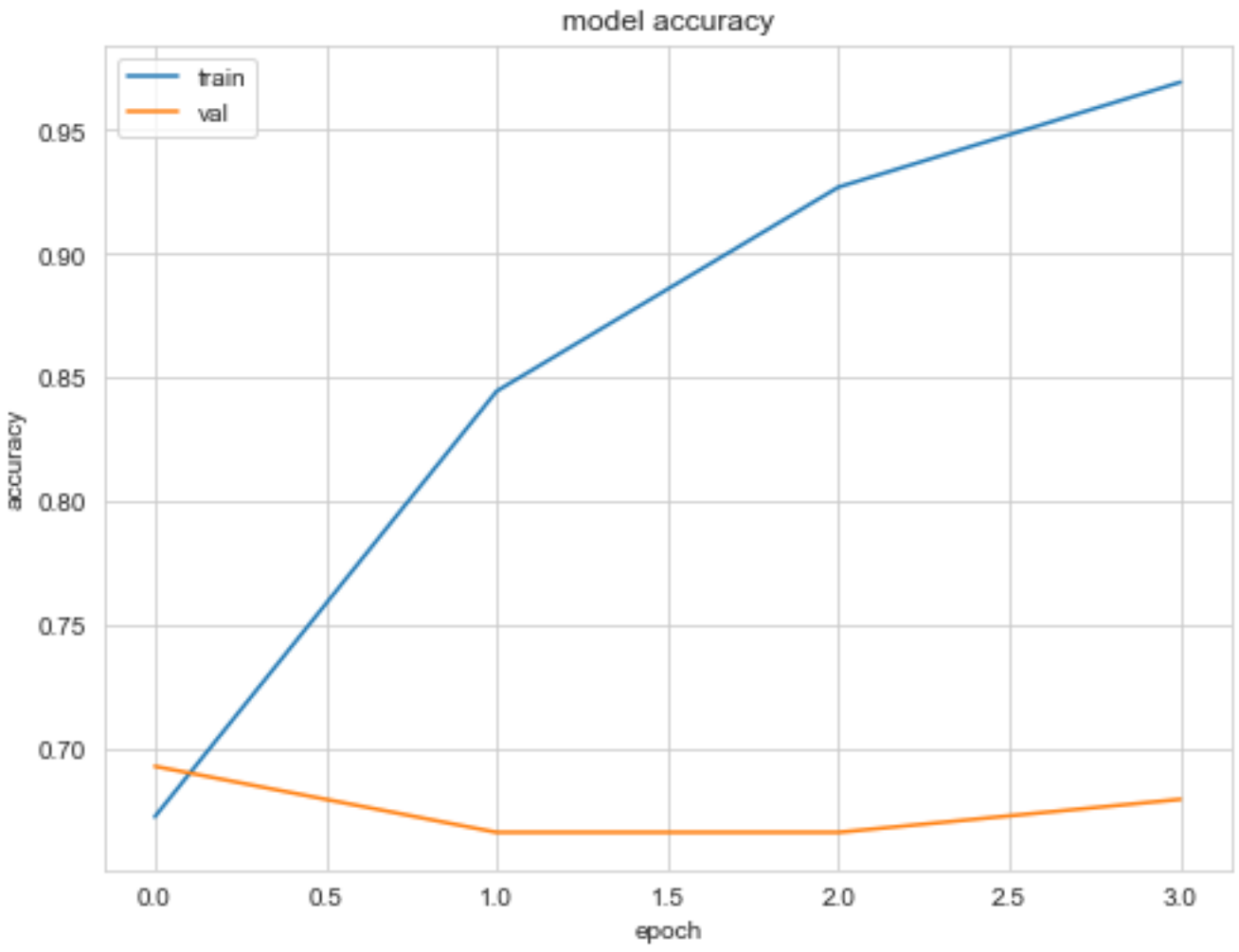
```
_____________________________________
Encoder-12-FeedForward (FeedFor (None, 128, 768)     4722432     Enc
oder-12-MultiHeadSelfAttention
_____________________________________
Encoder-12-FeedForward-Dropout  (None, 128, 768)     0           Enc
oder-12-FeedForward[0][0]
_____________________________________
Encoder-12-FeedForward-Add (Add (None, 128, 768)     0           Enc
oder-12-MultiHeadSelfAttention

Encoder-12-FeedForward-Dropout[0]
_____________________________________
Encoder-12-FeedForward-Norm (La (None, 128, 768)     1536        Enc
oder-12-FeedForward-Add[0][0]
_____________________________________
Extract (Extract)               (None, 768)          0           Enc
oder-12-FeedForward-Norm[0][0]
_____________________________________
NSP-Dense (Dense)               (None, 768)          590592      Ext
ract[0][0]
_____________________________________
dense_4 (Dense)                 (None, 4)            3076        NSP
-Dense[0][0]
==================================================================
==============================
Total params: 109,190,404
Trainable params: 109,190,404
Non-trainable params: 0
_____________________________________
_____________________________________
```

Plot the history for accuracy of Training/ Validation

In [303]:

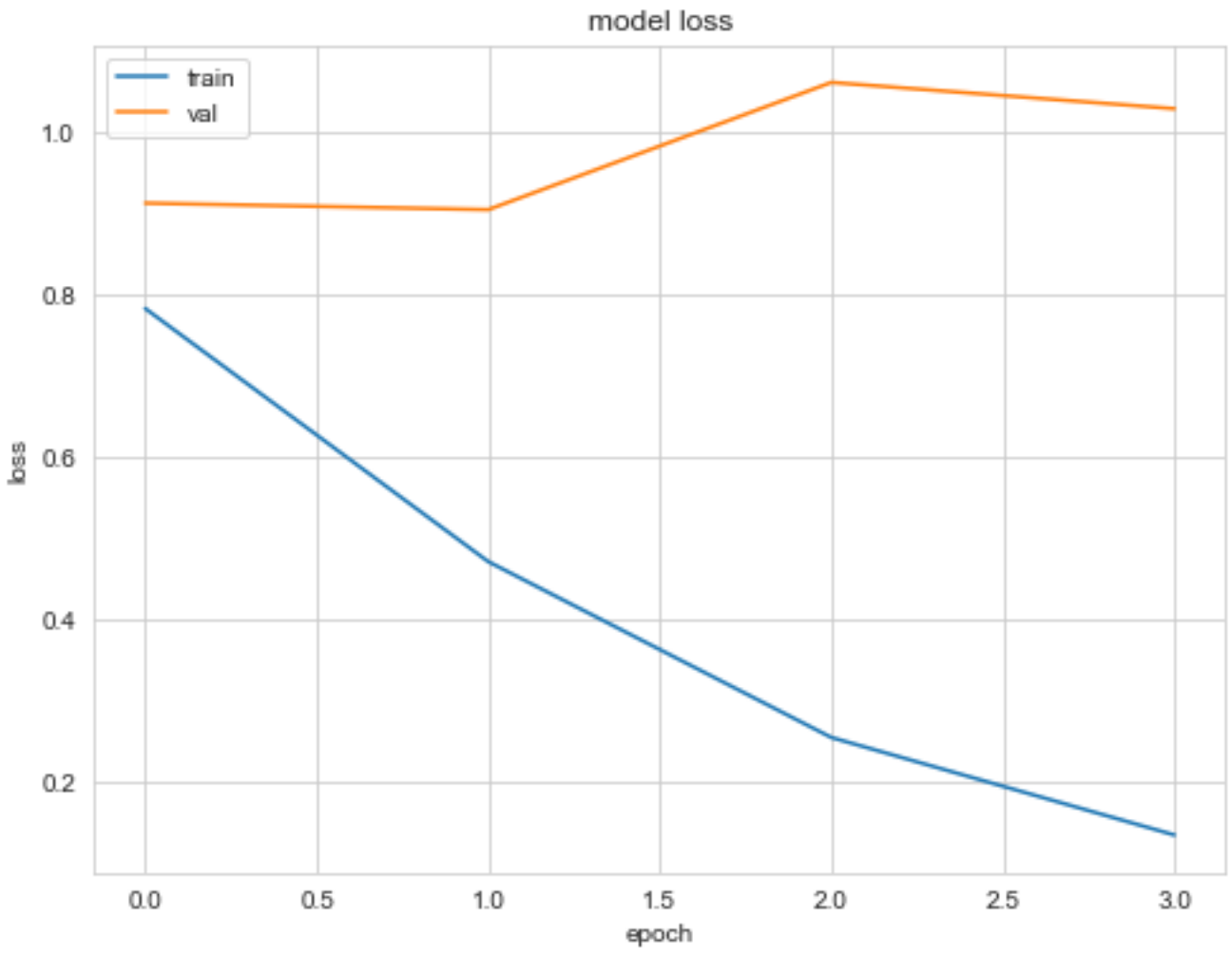
```
plt.figure(figsize=(8,6))
plt.plot(history.history['sparse_categorical_accuracy'])
plt.plot(history.history['val_sparse_categorical_accuracy'])
plt.title('model accuracy')
plt.xlabel('epoch')
plt.ylabel('accuracy')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```



Plot the history for loss of Training/ Validation

In [388]:

```
plt.figure(figsize=(8,6))
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```



Prediction on Test Data set of 50 example cases

In [ ]:

```
## Around 60 %
```

In [332]:

```
predicts = model.predict(X_test, verbose=True).argmax(axis=-1)
```

```
50/50 [==============================] - 60s 1s/step
```

In [356]:

```
print(np.sum(y_test == predicts) / y_test.shape[0])
```

```
0.58
```

In [ ]:

```
#### 0: SuicideWatch
#### 1 : depressed
#### 2 : happy
#### 3 :  selfimprovement
```

In [469]:

```
#  F1 Score is the weighted average of Precision and Recall. Therefore,
#this score takes both false positives and false negatives into account
#- F1 is usually more useful than accuracy, especially if you have an uneven cla
ss distribution.
```

In [363]:

```
from sklearn.metrics import precision_recall_fscore_support as score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import f1_score

precision, recall, fscore, support = score(y_test, predicts)
target_names = ['SuicideWatch', 'depressed', 'happy', 'selfimprovement']
print(classification_report(y_test, predicts, target_names=target_names))
# 'weighted' calculates de F1 score for each class independently but when it add
s them together uses a weight
#that depends on the number of true labels of each class: favouring the majority
class.
# 'micro' uses the global number of TP, FN, FP and calculates the F1 directly no
favouring any class in particular.
# 'macro average F1 score is the unweighted average of the F1-score over all the
classes in the multiclass case.
# It does not take into account the frequency of occurrence of the classes in th
e evaluation dataset.

f1_score(y_test, predicts, average='weighted')
```

```
                precision    recall  f1-score   support

   SuicideWatch       0.50      0.50      0.50        12
      depressed       0.20      0.14      0.17         7
          happy       1.00      0.53      0.70        15
selfimprovement       0.56      0.88      0.68        16

       accuracy                           0.58        50
      macro avg       0.56      0.51      0.51        50
   weighted avg       0.63      0.58      0.57        50
```

Out[363]:

```
0.5705655708731001
```

In [355]:

```
confusion_matrix(y_test, predicts,labels=[0,1,2,3])
```

Out[355]:

```
array([[ 6,  3,  0,  3],
       [ 4,  1,  0,  2],
       [ 1,  0,  8,  6],
       [ 1,  1,  0, 14]])
```

Practice cases Comparison with 500 training cases model : bert-tensorflow vs bert-keras

In [240]:

```
# [('I am so happy', array([-3.476168  , -3.7458189 , -0.09595221, -3.2981746 ],
dtype=float32),
# 'happy'),
# ('fuck', array([-0.7367987 , -2.458499  , -0.99514765, -2.7162235 ], dtype=flo
at32),
# 'SuicideWatch'),
# ('I feel nice',array([-3.3612282 , -3.5323107 , -0.11635606, -3.0810325 ], dty
pe=float32),
# 'happy'),
# ('I want to commit a suicide',  array([-0.30930468, -1.9100459 , -2.6896908 ,
-2.9943867 ], dtype=float32),
#'SuicideWatch'),
# ('I did self-improvement', array([-3.1551628, -3.093685 , -0.3177928, -1.69129
69], dtype=float32),
# 'happy'), --> This case needs to be fixed!!
# ('self-improvement', array([-2.2937117 , -2.275855  , -2.0193448 , -0.40998942
], dtype=float32),
# 'selfimprovement'),
# ('I felt happy yesterday but no more, now i want to die', array([-1.4680753 ,
-3.1400657 , -0.37962782, -3.1644907 ], dtype=float32),
# 'happy'), --> This case needs to be fixed!!
# ('I feel bad and want to die but I actually overcome this and become positive'
,array([-0.6128137, -2.0051608, -1.603159 , -2.1014626], dtype=float32),
#'SuicideWatch'), --> This case nes to be fixed!!
#('Absolutely fantastic!',array([-3.527619  , -3.9238014 , -0.09763478, -3.12631
25 ], dtype=float32),
# 'happy')] '''
```

In [237]:

```
# Encoding subreddit into 'subreddit_categorical_label' -> To use this column in
classification modeling
#### 0: SuicideWatch
#### 1 : depressed
#### 2 : happy
#### 3 :  selfimprovement
```

In [367]:

```
## Show a positive progress compare to previous model!
```

In [364]:

```
pred_sentences = [
    "I am so happy", #happy
    "fuck", # SuicideWatch
    "I feel nice", #selfimprovement  --> Can be considered as "happy"
    "I want to commit a suicide",#SuicideWatch
    "I did self-improvement", #selfimprovement
    "self-improvement",# selfimprovement
    "I felt happy yesterday but no more, now i want to die", #SuicideWatch
    "I feel bad and want to die but I actually overcome this and become positive
", #SuicideWatch --> Only this case needs to be fixed!
    "Absolutely fantastic!"#happy
]
```

In [365]:

```
indices= []
for example in pred_sentences:
    ids,segments =tokenizer.encode(example, max_len=SEQ_LEN)
    indices.append(ids)

example_x = [np.array(indices),np.zeros_like(np.array(indices))]
```

In [366]:

```
model.predict(example_x, verbose=True).argmax(axis=-1)
```

```
9/9 [==============================] - 12s 1s/step
```

Out[366]:

```
array([2, 0, 3, 0, 3, 3, 0, 0, 2])
```

In [ ]:

```

```

In [ ]:

```
###  Experiment
```

New training/test split for "Early stopping Functionality" test

In [375]:

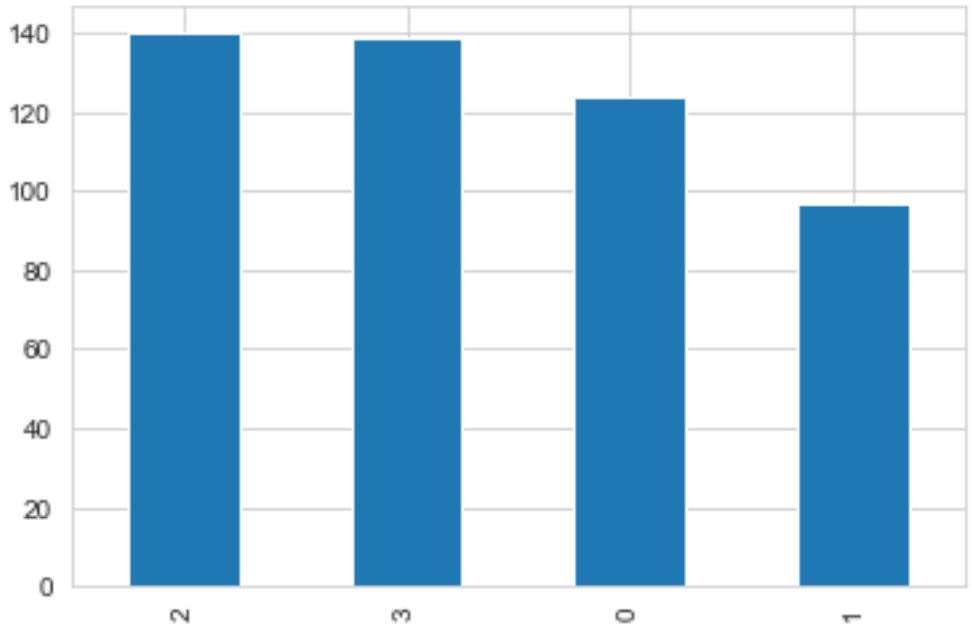
```
seed = 0
X_train, X_test, y_train, y_test = train_test_split(all_subreddit_df_list['title
_with_selftext_clean'], all_subreddit_df_list['subreddit_categorical_label'], \
                                                    test_size=0.33, random_state
=seed)
```

In [376]:

```
indices= []
for example in X_train.values[:500]:
    ids,segments =tokenizer.encode(example, max_len=SEQ_LEN)
    indices.append(ids)
X_train =[np.array(indices),np.zeros_like(np.array(indices))]
y_train = y_train.values[:500]
pd.Series(y_train).value_counts().plot(kind = 'bar')
```

Out[376]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1ed1f7f790>
```

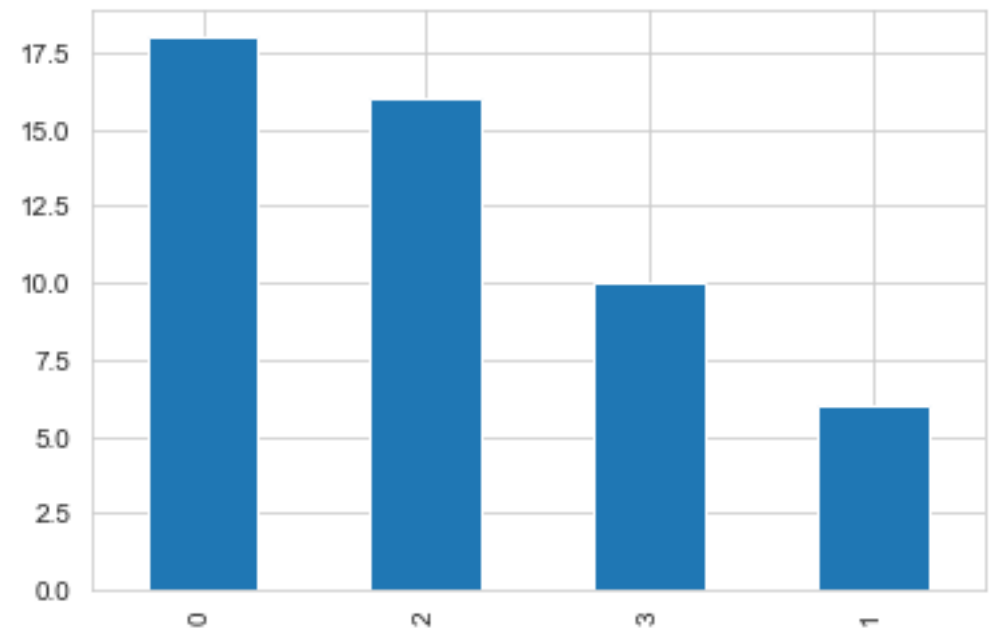


In [377]:

```
indices= []
for example in X_test.values[:50]:
    ids,segments =tokenizer.encode(example, max_len=SEQ_LEN)
    indices.append(ids)
X_test = [np.array(indices),np.zeros_like(np.array(indices))]
y_test = y_test.values[:50]
pd.Series(y_test).value_counts().plot(kind = 'bar')
```

Out[377]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1ed2059c10>
```



Early Stopping : to avoid overfitting in neural network

A problem with training neural networks is in the choice of the number of training epochs to use. Too many epochs can lead to overfitting of the training dataset, whereas too few may result in an underfit model.

Early stopping?

A method that allows you to specify an arbitrarily large number of training epochs and stop training once the model performance stops improving on the "Validation dataset".

This requires that a validation split should be provided to the fit() function and a "EarlyStopping" callback to specify performance measure on which performance will be monitored on validation split. Training will stop when the chosen performance measure stops improving. Once stopped, the "callback" will print the epoch number.

Patience argument?

Often, the first sign of no improvement may not be the best time to stop training and this is because the model may get slightly worse before getting much better sometimes.

We can account for this by adding a delay to the trigger in terms of the number of epochs on which we would like to see no improvement. This can be done by setting the “Patience” argument .The exact amount of patience will vary between models and problems. there a rule of thumb to make it 10% of number of epoch. For example, 1 for 10 epoch.

In [381]:

```
from keras.callbacks import EarlyStopping, ModelCheckpoint
```

We wanted to monitor the validation loss at each epoch and after the validation loss has not improved after two epochs, training is interrupted. However, since we set patience=2, we won’t get the best model, but the model two epochs after the best model.

So, An additional callback is required that will save the best model observed during training for later use. This is the

ModelCheckpoint callback.

In [382]:

```
#option1 :
history_new= model.fit(
    X_train,
    y_train,
    validation_split = 0.15,
    shuffle =True,
    epochs=5,
    batch_size=BATCH_SIZE,
    callbacks = [tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=2
, verbose =1),\
                         ModelCheckpoint(filepath='best_model.h5', monitor='val_lo
ss', save_best_only=True)]
)
```

```
Train on 425 samples, validate on 75 samples
Epoch 1/5
425/425 [==============================] - 4104s 10s/step - loss: 0.
3565 - sparse_categorical_accuracy: 0.8729 - val_loss: 0.7956 - val_
sparse_categorical_accuracy: 0.6800
Epoch 2/5
425/425 [==============================] - 1971s 5s/step - loss: 0.1
933 - sparse_categorical_accuracy: 0.9341 - val_loss: 0.8428 - val_s
parse_categorical_accuracy: 0.7333
Epoch 3/5
425/425 [==============================] - 1942s 5s/step - loss: 0.1
010 - sparse_categorical_accuracy: 0.9741 - val_loss: 1.1433 - val_s
parse_categorical_accuracy: 0.6800
Epoch 00003: early stopping
```

In [383]:

```
# list all data in history
history_new.history
```

Out[383]:

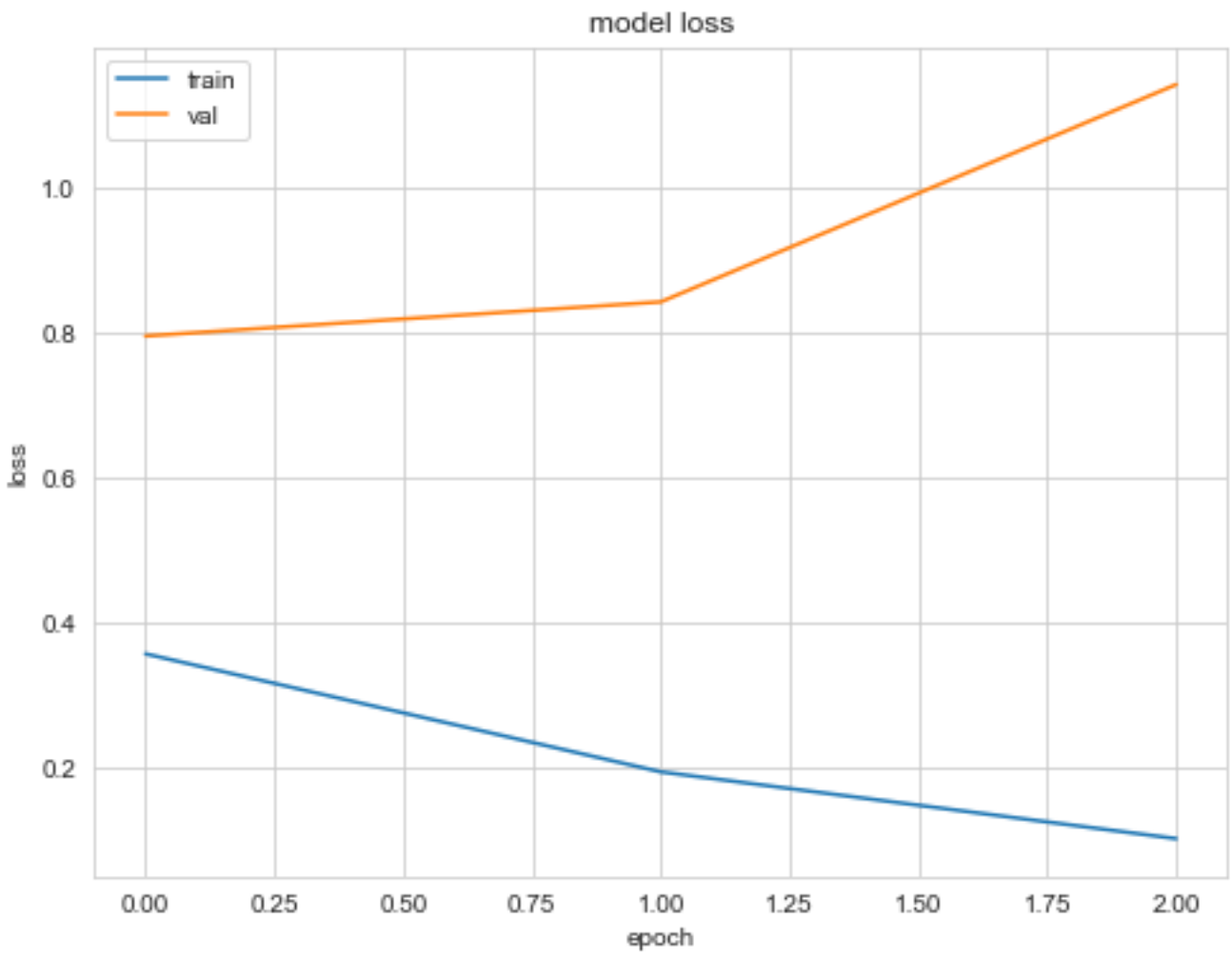
```
{'val_loss': [0.795638378461202, 0.8428328784306844, 1.1432575734456
38],
 'val_sparse_categorical_accuracy': [0.6800000023841858,
  0.7333333353201549,
  0.6799999984105428],
 'loss': [0.3564748824343962, 0.19333129896837123, 0.101043090399573
83],
 'sparse_categorical_accuracy': [0.8729411764705882,
  0.9341176470588235,
  0.9741176471990698]}
```

In [384]:

```
# summarize history for loss
```

In [387]:

```
plt.figure(figsize=(8,6))
plt.plot(history_new.history['loss'])
plt.plot(history_new.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```

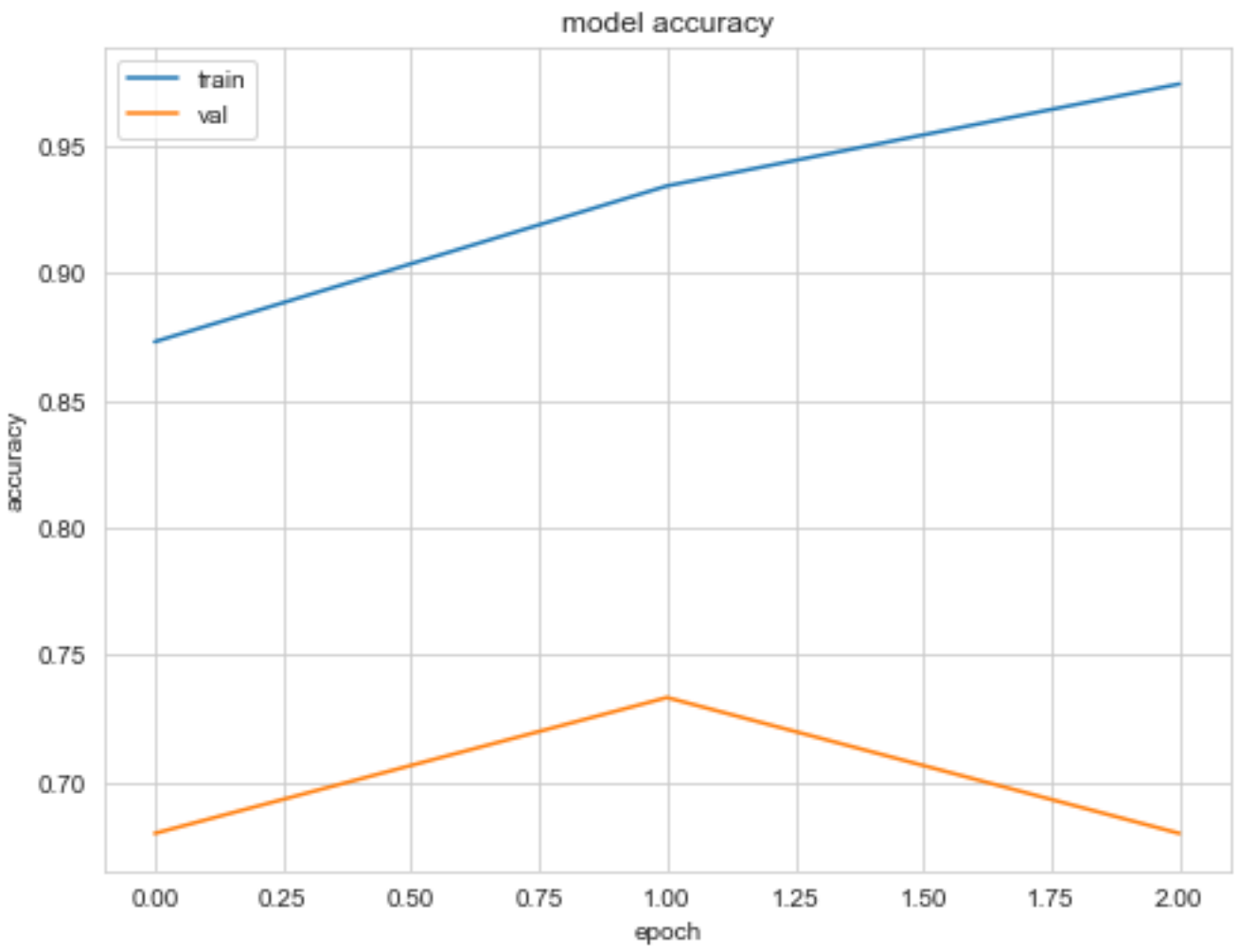


In [ ]:

```
# summarize history for accuracy
```

In [386]:

```
plt.figure(figsize=(8,6))
plt.plot(history_new.history['sparse_categorical_accuracy'])
plt.plot(history_new.history['val_sparse_categorical_accuracy'])
plt.title('model accuracy')
plt.xlabel('epoch')
plt.ylabel('accuracy')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```



Load the saved best model

In [390]:

```
from keras.models import load_model
from keras_bert import get_custom_objects
```

In [415]:

```
'''from keras_bert.layers import TokenEmbedding
from keras_pos_embd import PositionEmbedding
from keras_layer_normalization import LayerNormalization
from keras_multi_head import MultiHeadAttention
from keras_position_wise_feed_forward import FeedForward
from keras_transformer import gelu
CUSTOM_OBJECTS={'TokenEmbedding':TokenEmbedding,
'PositionEmbedding': PositionEmbedding,
'LayerNormalization': LayerNormalization,
'MultiHeadAttention': MultiHeadAttention,
'FeedForward': FeedForward,
'gelu': gelu}'''
```

Out[415]:

```
"from keras_bert.layers import TokenEmbedding\nfrom keras_pos_embd i
mport PositionEmbedding\nfrom keras_layer_normalization import Layer
Normalization\nfrom keras_multi_head import MultiHeadAttention\nfrom
keras_position_wise_feed_forward import FeedForward\nfrom keras_tran
sformer import gelu\nCUSTOM_OBJECTS={'TokenEmbedding':TokenEmbedding
, \n'PositionEmbedding': PositionEmbedding,\n'LayerNormalization': L
ayerNormalization,\n'MultiHeadAttention': MultiHeadAttention,\n'Feed
Forward': FeedForward,\n'gelu': gelu}"
```

Need to update our custom objects for loading our customed bert-model

In [413]:

```
custom_objects = get_custom_objects()
my_objects = {'RAdam': RAdam}
custom_objects.update(my_objects)
```

In [ ]:

```
## Successfully Load our best model!
```

In [414]:

```
saved_model = keras.models.load_model('best_model.h5', custom_objects=custom_obj
ects)
```

Prediction on Newly Assigned 50 Test Cases

& Previous vs Saved Model Performance Comparison

In [419]:

```
predicts_new = saved_model.predict(X_test, verbose=True).argmax(axis=-1)
print(np.sum(y_test == predicts_new) / y_test.shape[0])
```

```
50/50 [==============================] - 54s 1s/step
0.74
```

In [422]:

```
predicts_old = model.predict(X_test, verbose=True).argmax(axis=-1)
print(np.sum(y_test == predicts_old) / y_test.shape[0])
```

```
50/50 [==============================] - 60s 1s/step
0.74
```

In [430]:

```
precision, recall, fscore, support = score(y_test, predicts_old)
target_names = ['SuicideWatch', 'depressed', 'happy', 'selfimprovement']
print(classification_report(y_test, predicts_old, target_names=target_names))

# 'weighted' calculates de F1 score for each class independently but when it add
s them together uses a weight
# that depends on the number of true labels of each class: favouring the majorit
y class.

# 'micro' uses the global number of TP, FN, FP and calculates the F1 directly no
favouring any class in particular.
# 'macro average F1 score is the unweighted average of the F1-score over all the
classes in the multiclass case.
# It does not take into account the frequency of occurrence of the classes in th
e evaluation dataset.

f1_score(y_test, predicts_old, average='weighted')
```

```
                precision    recall  f1-score   support

   SuicideWatch       1.00      0.56      0.71        18
      depressed       0.43      0.50      0.46         6
          happy       0.83      0.94      0.88        16
selfimprovement       0.60      0.90      0.72        10

       accuracy                           0.74        50
      macro avg       0.72      0.72      0.69        50
   weighted avg       0.80      0.74      0.74        50
```

Out[430]:

```
0.7388804137039432
```

In [431]:

```
precision, recall, fscore, support = score(y_test, predicts_new)
target_names = ['SuicideWatch', 'depressed', 'happy', 'selfimprovement']
print(classification_report(y_test, predicts_new, target_names=target_names))
f1_score(y_test, predicts_new, average='weighted')
```

```
                precision    recall  f1-score   support

   SuicideWatch       0.92      0.61      0.73        18
      depressed       0.33      0.67      0.44         6
          happy       0.83      0.94      0.88        16
selfimprovement       0.88      0.70      0.78        10

       accuracy                           0.74        50
      macro avg       0.74      0.73      0.71        50
   weighted avg       0.81      0.74      0.76        50
```

Out[431]:

```
0.7552418300653595
```

In [ ]:

```
### Confusion Matrix comparison
```

In [425]:

```
confusion_matrix(y_test, predicts_old,labels=[0,1,2,3])
```

Out[425]:

```
array([[10,  3,  1,  4],
       [ 0,  3,  1,  2],
       [ 0,  1, 15,  0],
       [ 0,  0,  1,  9]])
```

In [426]:

```
confusion_matrix(y_test, predicts_new,labels=[0,1,2,3])
```

Out[426]:

```
array([[11,  6,  1,  0],
       [ 1,  4,  0,  1],
       [ 0,  1, 15,  0],
       [ 0,  1,  2,  7]])
```

In [ ]:

Same example case model prediction comparison

Case1 : Bert-tensorflow - previous version (Traing/val with 500 cases)

- Accuracy : 60-63%

In [481]:

```
'''[('I am so happy', array([-3.476168  , -3.7458189 , -0.09595221, -3.2981746 ]
, dtype=float32),
# 'happy'),
# ('fuck', array([-0.7367987 , -2.458499  , -0.99514765, -2.7162235 ], dtype=flo
at32),
# 'SuicideWatch'),
# ('I feel nice',array([-3.3612282 , -3.5323107 , -0.11635606, -3.0810325 ], dty
pe=float32),
# 'happy'),
# ('I want to commit a suicide',  array([-0.30930468, -1.9100459 , -2.6896908 ,
-2.9943867 ], dtype=float32),
#'SuicideWatch'),

# ('I did self-improvement', array([-3.1551628, -3.093685 , -0.3177928, -1.69129
69], dtype=float32),
# 'happy'), --> This case needs to be fixed!!

# ('self-improvement', array([-2.2937117 , -2.275855  , -2.0193448 , -0.40998942
], dtype=float32),
# 'selfimprovement'),

# ('I felt happy yesterday but no more, now i want to die', array([-1.4680753 ,
-3.1400657 , -0.37962782, -3.1644907 ], dtype=float32),
# 'happy'), --> This case needs to be fixed!!

# ('I feel bad and want to die but I actually overcome this and become positive'
,array([-0.6128137, -2.0051608, -1.603159 , -2.1014626], dtype=float32),
#'SuicideWatch'), --> This case might need to be fixed!!

#('Absolutely fantastic!',array([-3.527619  , -3.9238014 , -0.09763478, -3.12631
25 ], dtype=float32),
# 'happy')] '''
```

Out[481]:

"[('I am so happy', array([-3.476168 , -3.7458189 , -0.09595221, -3
.2981746], dtype=float32), \n# 'happy'),\n# ('fuck', array([-0.7367
987 , -2.458499 , -0.99514765, -2.7162235], dtype=float32),\n# 'Su
icideWatch'),\n# ('I feel nice',array([-3.3612282 , -3.5323107 , -0.
11635606, -3.0810325], dtype=float32),\n# 'happy'),\n# ('I want to
commit a suicide', array([-0.30930468, -1.9100459 , -2.6896908 , -2
.9943867], dtype=float32), \n#'SuicideWatch'),\n\n# ('I did self-im
provement', array([-3.1551628, -3.093685 , -0.3177928, -1.6912969],
dtype=float32),\n# 'happy'), --> This case needs to be fixed!!\n\n#
('self-improvement', array([-2.2937117 , -2.275855 , -2.0193448 , -
0.40998942], dtype=float32),\n# 'selfimprovement'),\n\n# ('I felt ha
ppy yesterday but no more, now i want to die', array([-1.4680753 , -
3.1400657 , -0.37962782, -3.1644907], dtype=float32),\n# 'happy'),
--> This case needs to be fixed!! \n\n# ('I feel bad and want to di
e but I actually overcome this and become positive',array([-0.612813
7, -2.0051608, -1.603159 , -2.1014626], dtype=float32),\n#'SuicideWa
tch'), --> This case might need to be fixed!!\n\n#('Absolutely fanta
stic!',array([-3.527619 , -3.9238014 , -0.09763478, -3.1263125], d
type=float32), \n# 'happy')] "

Case2 : Bert-without Stopping (Traing/val with 500 cases) - Accuracy : 60-73%

In [440]:

```
'''pred_sentences = [
    "I am so happy", #happy
    "fuck", # SuicideWatch
    "I feel nice", #selfimprovement  --> Can be considered as "happy"
    "I want to commit a suicide",#SuicideWatch
    "I did self-improvement", #selfimprovement
    "self-improvement",# selfimprovement
    "I felt happy yesterday but no more, now i want to die", #SuicideWatch
    "I feel bad and want to die but I actually overcome this and become positive
",
    #SuicideWatch --> This case needs to be fixed!
    "Absolutely fantastic!"#happy
]'''
```

Out[440]:

```
'pred_sentences = [\n    "I am so happy", #happy\n    "fuck", # Suic
ideWatch\n    "I feel nice", #selfimprovement  --> Can be considered
as "happy"\n    "I want to commit a suicide",#SuicideWatch\n    "I d
id self-improvement", #selfimprovement\n    "self-improvement",# sel
fimprovement\n    "I felt happy yesterday but no more, now i want to
die", #SuicideWatch \n    "I feel bad and want to die but I actually
overcome this and become positive", #SuicideWatch --> Only this case
needs to be fixed!\n    "Absolutely fantastic!"#happy\n]'
```

Case3 : Bert-with EarlyStopping - prevent over/under-fitting model

(Traing/val with 500 cases) - Accuracy : 73%

In [432]:

```
pred_sentences = [
    "I am so happy", #happy
    "fuck", # SuicideWatch
    "I feel nice", #happy --> Nice improvement case of saved best model compare
to previous 4 epoch model
    "I want to commit a suicide",#SuicideWatch
    "I did self-improvement", #selfimprovement
    "self-improvement",# selfimprovement
    "I felt happy yesterday but no more, now i want to die", #SuicideWatch

    "I feel bad and want to die but I actually overcome this and become positive
",
    #SuicideWatch --> Only this case might need to be fixed!

    "Absolutely fantastic!"#happy
]
```

In [417]:

```
indices= []
for example in pred_sentences:
    ids,segments =tokenizer.encode(example, max_len=SEQ_LEN)
    indices.append(ids)
example_x = [np.array(indices),np.zeros_like(np.array(indices))]
saved_model.predict(example_x, verbose=True).argmax(axis=-1)
```

```
9/9 [==============================] - 33s 4s/step
```

Out[417]:

```
array([2, 0, 2, 0, 3, 3, 0, 0, 2])
```

Case4 : Bert-with EarlyStopping - prevent over/under-fitting model

(Traing/val with 1000 cases) - Accuracy : 77%

In [ ]:

```
pred_sentences = [
    "I am so happy", #happy
    "fuck", # SuicideWatch
    "I feel nice", #happy
    "I want to commit a suicide",#SuicideWatch
    "I did self-improvement", #selfimprovement
    "self-improvement",# selfimprovement
    "I felt happy yesterday but no more, now i want to die", #SuicideWatch

    "I feel bad and want to die but I actually overcome this and become happy",
    #SuicideWatch --> Only this case might need to be fixed!

    "Absolutely fantastic!"#happy
]
```

In [ ]:

In [ ]:

Final training/test split for 1000 training case model test

In [451]:

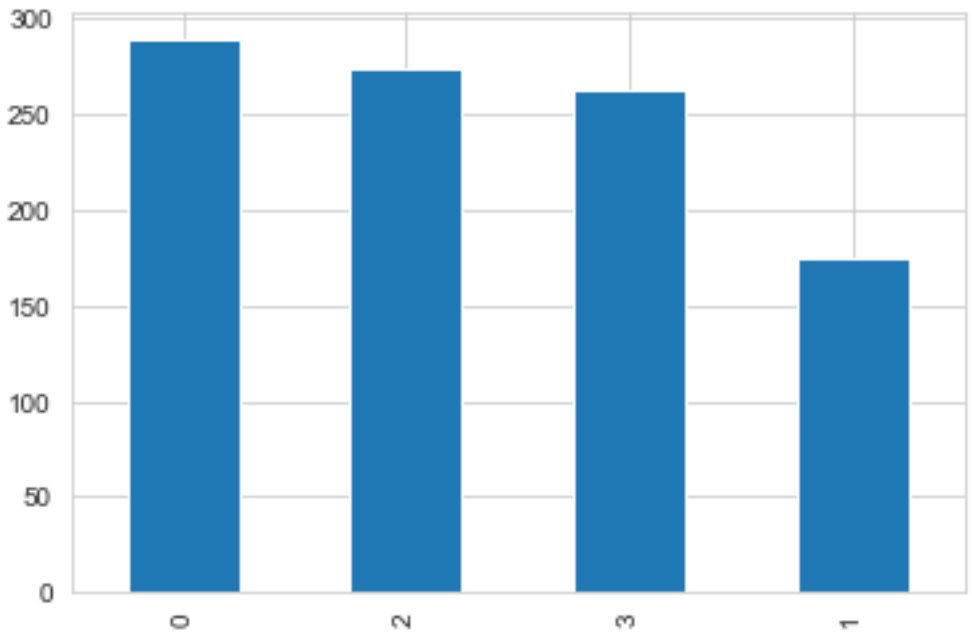
```
seed = 10
X_train, X_test, y_train, y_test = train_test_split(all_subreddit_df_list['title
_with_selftext_clean'], all_subreddit_df_list['subreddit_categorical_label'], \
                                                    test_size=0.33, random_state
=seed)
```

In [452]:

```
indices= []
for example in X_train.values[:1000]:
    ids,segments =tokenizer.encode(example, max_len=SEQ_LEN)
    indices.append(ids)
X_train =[np.array(indices),np.zeros_like(np.array(indices))]
y_train = y_train.values[:1000]
pd.Series(y_train).value_counts().plot(kind = 'bar')
```

Out[452]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1b18866350>
```

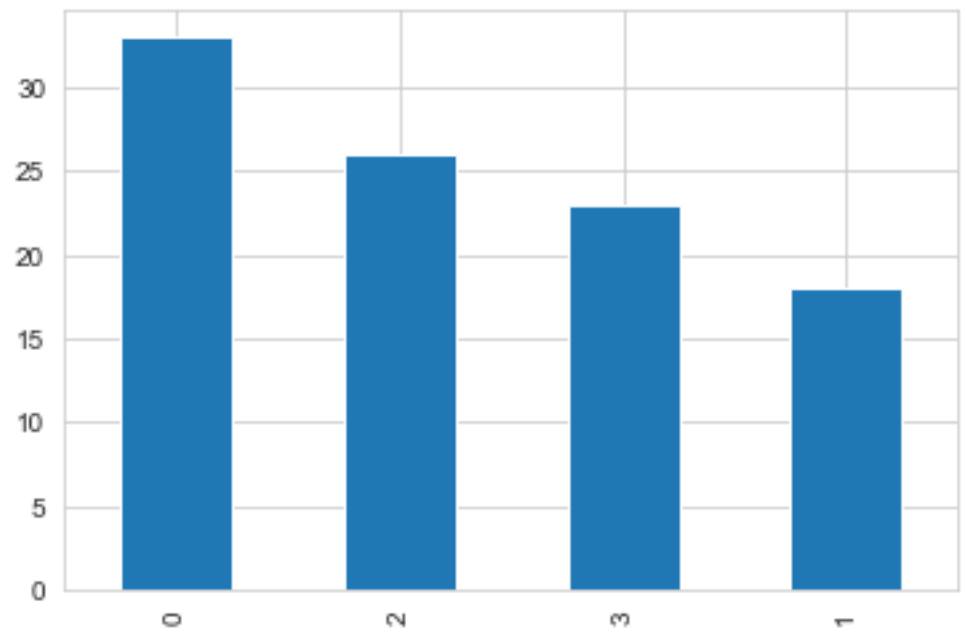


In [453]:

```
indices= []
for example in X_test.values[:100]:
    ids,segments =tokenizer.encode(example, max_len=SEQ_LEN)
    indices.append(ids)
X_test = [np.array(indices),np.zeros_like(np.array(indices))]
y_test = y_test.values[:100]
pd.Series(y_test).value_counts().plot(kind = 'bar')
```

Out[453]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1ecfe4d110>
```



In [460]:

```
history_new1= model.fit(
    X_train,
    y_train,
    validation_split = 0.15,
    shuffle =True,
    epochs=5,
    batch_size=BATCH_SIZE,
    callbacks = [tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=2
, verbose =1),\
                         ModelCheckpoint(filepath='best_model_1000.h5', monitor='v
al_loss', save_best_only=True)]
)
```

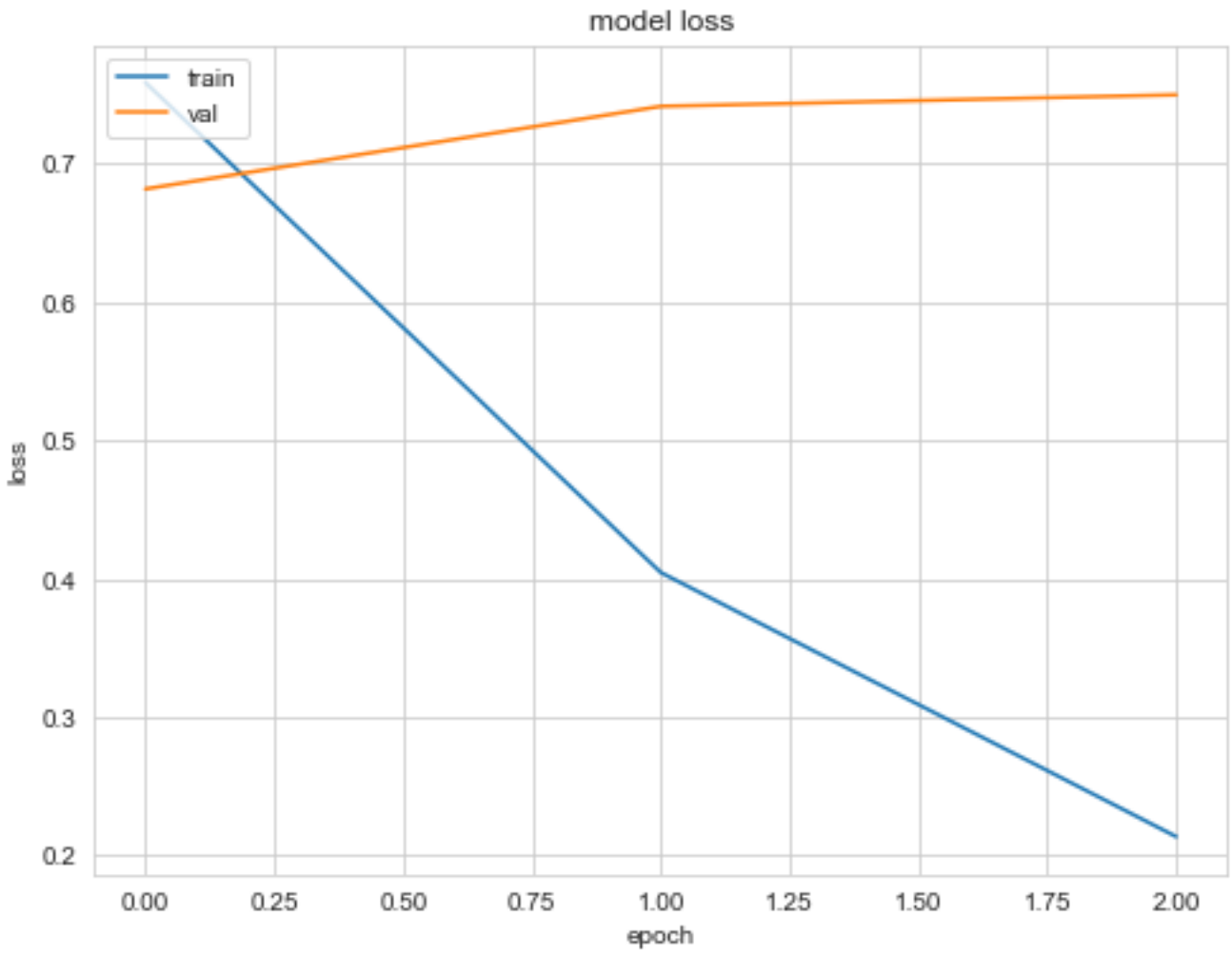
```
Train on 850 samples, validate on 150 samples
Epoch 1/5
850/850 [==============================] - 5067s 6s/step - loss: 0.7
585 - sparse_categorical_accuracy: 0.7106 - val_loss: 0.6816 - val_s
parse_categorical_accuracy: 0.7267
Epoch 2/5
850/850 [==============================] - 5106s 6s/step - loss: 0.4
044 - sparse_categorical_accuracy: 0.8612 - val_loss: 0.7413 - val_s
parse_categorical_accuracy: 0.7267
Epoch 3/5
850/850 [==============================] - 4640s 5s/step - loss: 0.2
135 - sparse_categorical_accuracy: 0.9259 - val_loss: 0.7497 - val_s
parse_categorical_accuracy: 0.7467
Epoch 00003: early stopping
```

In [ ]:

```
# summarize history for loss
```

In [461]:

```
plt.figure(figsize=(8,6))
plt.plot(history_new1.history['loss'])
plt.plot(history_new1.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```

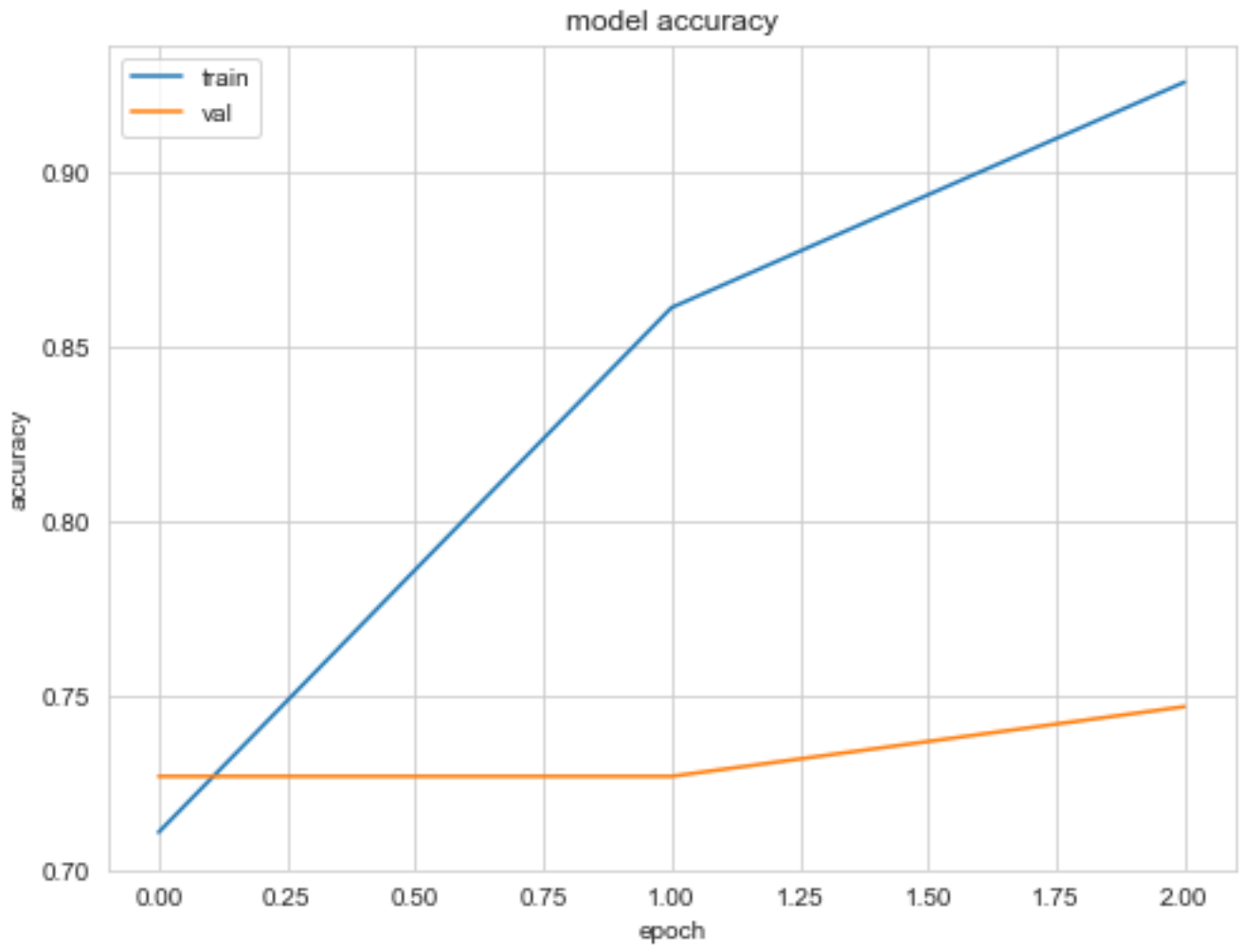


In [ ]:

```
# summarize history for accuracy
```

In [462]:

```
plt.figure(figsize=(8,6))
plt.plot(history_new1.history['sparse_categorical_accuracy'])
plt.plot(history_new1.history['val_sparse_categorical_accuracy'])
plt.title('model accuracy')
plt.xlabel('epoch')
plt.ylabel('accuracy')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```



Test accuracy for 100 cases : 77%

Load the saved best model

In [466]:

```
saved_model = keras.models.load_model('best_model_1000.h5', custom_objects=custo
m_objects)
```

In [467]:

```
predicts_new1 = saved_model.predict(X_test, verbose=True).argmax(axis=-1)
print(np.sum(y_test == predicts_new1) / y_test.shape[0])
```

```
100/100 [==============================] - 154s 2s/step
0.77
```

In [473]:

```
precision, recall, fscore, support = score(y_test, predicts_new1)
target_names = ['SuicideWatch', 'depressed', 'happy', 'selfimprovement']
print(classification_report(y_test, predicts_new1, target_names=target_names))

# 'weighted' calculates de F1 score for each class independently but when it add
s them together uses a weight
# that depends on the number of true labels of each class: favouring the majorit
y class.

# 'micro' uses the global number of TP, FN, FP and calculates the F1 directly no
favouring any class in particular.
# 'macro average F1 score is the unweighted average of the F1-score over all the
classes in the multiclass case.
# It does not take into account the frequency of occurrence of the classes in th
e evaluation dataset.
```

```
f1_score(y_test, predicts_new1, average='weighted')
```

```
                precision    recall  f1-score   support

   SuicideWatch       0.69      0.88      0.77        33
      depressed       0.44      0.22      0.30        18
          happy       0.89      0.96      0.93        26
selfimprovement       0.90      0.83      0.86        23

       accuracy                           0.77       100
      macro avg       0.73      0.72      0.71       100
   weighted avg       0.75      0.77      0.75       100
```

Out[473]:

```
0.7479104377104376
```

In [478]:

```
pred_sentences = [
    "I am so happy", #happy
    "fuck", # SuicideWatch
    "I feel nice", #happy
    "I want to commit a suicide",#SuicideWatch
    "I did self-improvement", #selfimprovement
    "self-improvement",# selfimprovement
    "I felt happy yesterday but no more, now i want to die", #SuicideWatch

    "I feel bad and want to die but I actually overcome this and become happy",
    #SuicideWatch --> Only this case might need to be fixed!

    "Absolutely fantastic!"#happy
]
```

In [479]:

```
indices= []
for example in pred_sentences:
    ids,segments =tokenizer.encode(example, max_len=SEQ_LEN)
    indices.append(ids)
example_x = [np.array(indices),np.zeros_like(np.array(indices))]
saved_model.predict(example_x, verbose=True).argmax(axis=-1)
```

```
9/9 [==============================] - 9s 1s/step
```

Out[479]:

```
array([2, 0, 2, 0, 3, 3, 0, 2, 2])
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```
```

In [ ]:

```
```

In [ ]:

```
```

In [ ]:

```
```

In [ ]:

```
```

Earlystopping other option using "Training accuracy" treshold :

In [ ]:

```
#option2:
#stop training using callback
#https://towardsdatascience.com/neural-network-with-tensorflow-how-to-stop-training-using-callback-5c8d575c18a9

# Implement callback function to stop training
# when accuracy reaches ACCURACY_THRESHOLD
ACCURACY_THRESHOLD = 0.95

class myCallback(atf.keras.callbacks.Callback):
    def on_epoch_end(self, epoch, logs={}):
        if(logs.get('acc') > ACCURACY_THRESHOLD):
            print("\n\n Training Reached %2.2f%% accuracy, \
            so now we stopping training!!" %(ACCURACY_THRESHOLD*100))
            self.model.stop_training = True

# Instantiate a callback object
callbacks = myCallback()

# and then
model.fit(x_train, y_train, epochs=5, callbacks=[callbacks])
```

In [ ]:

```
```

In [ ]:

In [ ]:

```
## one possible consideration for next_step
```

In [ ]:

```
'''
We can also build our model's layer with bert_layer in Keras -> we can try this
approach
model = tf.keras.Sequential([
        tf.keras.layers.Input(shape=(max_seq_length,), dtype='int32', name='inpu
t_ids'),
        bert_layer,
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(256, activation=tf.nn.relu),
        tf.keras.layers.Dropout(0.1),
        tf.keras.layers.Dense(256, activation=tf.nn.relu),
        tf.keras.layers.Dropout(0.1),
        tf.keras.layers.Dense(classes, activation=tf.nn.softmax)
    ])

    model.build(input_shape=(None, max_seq_length))
    '''
```

In [ ]:

```
#<Reference>
#https://blog.ekbana.com/fine-tuning-bert-for-text-classification-20news-group-c
lassification-53a55dc09738
#https://colab.research.google.com/drive/1YSfscbb-g92m1vkYxY4IOVMWMfgfLLJDe
```

In [ ]: