In [0]:

```
import tarfile
tar = tarfile.open('/content/emotions.tar.gz')
tar.extractall()
tar.close()
```

In [2]:

```
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")

import sqlite3
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
from nltk.stem.porter import PorterStemmer

import re
# Tutorial about Python regular expressions: https://pymotw.com/2/re/
import string
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.stem.wordnet import WordNetLemmatizer

from gensim.models import Word2Vec
from gensim.models import KeyedVectors
import pickle

from tqdm import tqdm
import os
from collections import Counter


# Credits: https://machinelearningmastery.com/sequence-classification-lstm-recurrent-neural-networ
ks-python-keras/
# LSTM for sequence classification in the IMDB dataset
import numpy
from keras.datasets import imdb
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers.embeddings import Embedding
from keras.preprocessing import sequence
# fix random seed for reproducibility
numpy.random.seed(7)
from numpy import array
from numpy import asarray
from numpy import zeros
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Flatten

from keras.layers import Embedding
from keras.layers import Dense,LSTM
from keras import Input
import numpy as np
np.random.seed(0)
from keras.models import Model
from keras.layers import Dense, Input, Dropout, LSTM, Activation,Reshape
```

```
from keras.layers.embeddings import Embedding
from keras.preprocessing import sequence
from keras.initializers import glorot_uniform
import keras
from keras.models import load_model, Model
from keras.layers import Dense, Activation, Dropout, Input, LSTM, Reshape, Lambda, RepeatVector
from keras.initializers import glorot_uniform
from keras.utils import to_categorical
from keras.optimizers import Adam
from keras import backend as K
from sklearn.metrics import roc_auc_score
from sklearn.datasets import make_classification
from keras.models import Sequential
import tensorflow as tf
from sklearn.metrics import roc_auc_score

from keras.layers import Dense
from keras.utils import np_utils
from keras.callbacks import Callback, EarlyStopping
```

```
Using TensorFlow backend.
```

The default version of TensorFlow in Colab will soon switch to TensorFlow 2.x.
We recommend you upgrade now or ensure your notebook will continue to use TensorFlow 1.x via the `%tensorflow_version 1.x` magic: more info.

In [0]:

```
from numpy.random import seed
seed(1)
from tensorflow import set_random_seed
set_random_seed(2)
```

In [0]:

```
def list_data(path):
  list_data= []
  for i in os.listdir(path):
    data = pd.read_csv( path + '/' + str(i), sep=",", header=None)
    list_data.append(data[0][0])
  return list_data
```

In [0]:

```
angry_list  = list_data('/content/emotions/angry')
```

In [0]:

```
happy_list  = list_data('/content/emotions/happy')
```

In [0]:

```
sad_list  = list_data('/content/emotions/sad')
```

In [0]:

```
neutral_list  = list_data('/content/emotions/neutral')
```

In [0]:

```
all_words=set()
for line in sad_list:
    for word in line.split(' '):
        if word not in all_words:
            all_words.add(word)

for line in neutral_list:
    for word in line.split(' '):
```

```
        if word not in all_words:
            all_words.add(word)

for line in happy_list:
    for word in line.split(' '):
        if word not in all_words:
            all_words.add(word)

for line in angry_list:
    for word in line.split(' '):
        if word not in all_words:
            all_words.add(word)
```

Adding all text into a single List

In [0]:

```
list_sentences = sad_list + neutral_list + happy_list + angry_list
```

In [0]:

```
df = pd.DataFrame()
```

In [0]:

```
output_y = []

for i in range(len(sad_list)):
  output_y.append(1)
for i in range(len(neutral_list)):
  output_y.append(2)
for i in range(len(happy_list)):
  output_y.append(3)
for i in range(len(angry_list)):
  output_y.append(4)
```

Shuffling the Dataframe

In [0]:

```
df['text'] = list_sentences
```

In [0]:

```
df['output'] =output_y
from sklearn.utils import shuffle
df = shuffle(df)
```

In [0]:

```
list_sentences = df['text']
```

In [0]:

```
output_y = df['output']
```

In [17]:

```
from sklearn.preprocessing import OneHotEncoder
onehot_encoder = OneHotEncoder()

y_train = np.array(output_y)
y_encoded = onehot_encoder.fit_transform(y_train.reshape(-1,1)).toarray()
print('outputs_encoded.shape after One Hot Encode', y_encoded.shape)
```

```
outputs encoded.shape after One Hot Encode (513, 4)
```

In [18]:

```
list_sentences[400:405]
```

Out[18]:

```
259                                 तुम बढ़िया हो यार
333                                          थैंक्स
369    यीपी ! ! ! मेरा फ़्लाइट आखिरकार बुक हो गया
321                      आ हा ! सारथी तुम बेस्ट हो
183          मेरे अकाउंट में कितना बैलेंस बचा है
Name: text, dtype: object
```

remove special characters

In [0]:

```
bad_chars = [';', ':', '!', "*"]
for j in range(len(list_sentences)):

  for i in  list_sentences[j].split() :
      if i in bad_chars:
        x = list_sentences[j].replace(i,'')
        list_sentences[j] =x
```

In [20]:

```
from keras import backend as K

K.clear_session()
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:107: The name tf.reset_default_graph is deprecated. P
lease use tf.compat.v1.reset_default_graph instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:111: The name tf.placeholder_with_default is
deprecated. Please use tf.compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:66: The name tf.get_default_graph is deprecated. Plea
se use tf.compat.v1.get_default_graph instead.
```

In [21]:

```
! pip install uniseg
```

```
Requirement already satisfied: uniseg in /usr/local/lib/python3.6/dist-packages (0.7.1)
```

In [0]:

```
from uniseg.graphemecluster import grapheme_clusters
```

Convert Word to Characters

In [65]:

```
# Combining all the above statemennts
from tqdm import tqdm
char_sentence = []
# tqdm is for printing the status bar
for sentance in tqdm(list_sentences.values):
  char_list = [char for char in grapheme_clusters(sentance)]

  char_sentence.append(char_list)
```

```
100%|██████████| 513/513 [00:00<00:00, 3355.13it/s]
```

In [0]:

```
char_sentence = pd.Series(char_sentence)
```

In [67]:

```
char_sentence
```

Out[67]:

```
0       [मे, रे,  , को,  , तु, म्, हा, रा,  , स, र्, व...
1                  [शु, क्, रि, या,  , तु, म्, हा, रा,  ]
2              [ये,  , हो, ट, ल,  , अ, च्, छा,  , है,  ]
3       [स, ब, से,  , क, म,  , टा, इ, म,  , ल, ग, ने, ...
4       [ए, प्, प,  , बा, र,  , बा, र,  , फ, स,  , क्,...
                                ...
508     [वा, क, ई,  , ता, री, फ,  , के,  , का, बि, ल, ...
509     [तु, म,  , बु, किं, ग,  , क, र, ने,  , में,  ,...
510     [लं, ड,  , जै, सा,  , का, म,  , कि, या,  , तु,...
511      [तु, म,  , ब, हु, त,  , स्, मा, र्, ट,  , हो,  ]
512     [ट्, रै, न,  , टि, क, ट,  , अ, प, ने,  , आ, प,...
Length: 513, dtype: object
```

Tokenizer for char's

In [68]:

```
maxLen = len(max(char_sentence, key=len))
maxLen
# total no of characters in all sentences
```

Out[68]:

```
44
```

fit and transform Train Char-tokenizer using first 400 words

In [69]:

```
from keras.preprocessing.text import Tokenizer

t_char = Tokenizer()
t_char.fit_on_texts(char_sentence[:400])
vocab_size = len(t_char.word_index) + 1
# integer encode the documents
encoded_docs_train_char = t_char.texts_to_sequences(char_sentence[:400])
print(encoded_docs_train_char)
# pad documents to a max length of 4 words
max_length = 45
padded_docs_train_char = sequence.pad_sequences(encoded_docs_train_char, maxlen=max_length,
padding='post')
print(padded_docs_train_char)
```

```
[[28, 23, 1, 63, 1, 26, 54, 33, 24, 1, 5, 41, 76, 5, 1, 88, 11, 45, 1, 21, 61, 81, 1, 8, 29, 1, 10
, 55], [163, 12, 69, 4, 1, 26, 54, 33, 24, 1], [39, 1, 14, 15, 10, 1, 21, 61, 81, 1, 9, 1], [5, 6,
16, 1, 3, 7, 1, 96, 20, 7, 1, 10, 17, 27, 1, 25, 97, 1, 64, 23, 8, 1, 6, 36, 65], [13, 56, 18, 1,
34, 2, 1, 34, 2, 1, 114, 5, 1, 12, 98, 1, 2, 33, 1, 9, 1], [20, 5, 1, 7, 42, 27, 1, 73, 10, 1, 99,
147, 107, 1, 20, 11, 27, 1, 58, 16, 1, 46, 1], [1, 59, 3, 15, 1, 3, 82, 114, 41, 7, 1, 9, 1, 12, 4
, 1], [22, 107, 1, 12, 4, 1, 3, 2, 45, 1, 5, 7, 83, 1, 8, 29, 1, 22, 1, 2, 33, 1, 9, 1], [38, 37,
25, 6, 1], [14, 15, 10, 1, 63, 1, 66, 11, 100, 1, 23, 182, 17, 1, 89, 97, 1, 9, 1], [14, 15, 10, 1
, 84, 1, 183, 3, 1, 42, 1, 60], [26, 7, 27, 1, 84, 1, 3, 90, 10, 1, 3, 2, 1, 47, 4, 1], [184, 148,
7, 1, 5, 7, 57, 1, 3, 6, 1, 11, 3, 1, 9, 1, 69, 80, 41, 37, 1, 3, 2, 27, 1, 49, 1], [70, 38, 20, 1
5, 1, 59, 3, 15, 1, 21, 164, 50, 6, 10, 1, 9, 1, 12, 4, 1], [7, 31, 31, 1, 3, 2, 27, 1, 40, 1, 91,
13, 1, 163, 12, 69, 4, 1], [149, 15, 1, 4, 2, 1, 150, 2, 1, 16, 1, 115, 8, 51, 5, 1, 165, 31, 1, 1
4, 1, 17, 13, 1], [14, 15, 10, 1, 46, 1, 44, 76, 220, 17, 1, 221, 10, 1, 92, 2, 1, 34, 2, 1, 8, 29
, 1, 60, 1, 4, 2, 1], [13, 3, 1, 77, 1, 5, 42, 1, 131, 37, 1, 8, 29, 1, 6, 36, 43, 1, 26, 7, 27],
```

[28, 24, 1, 64, 93, 8, 1, 59, 3, 15, 1, 48, 3, 1, 14, 1, 17, 4, 1, 25, 65, 1, 1, 1, 1], [20, 11, 4
5, 1, 132, 82, 31, 2, 1, 14, 15, 10, 1, 9, 1], [35, 6, 1, 184, 31, 2, 1, 16, 1, 19, 108, 1, 17, 82
, 94, 1, 67, 1], [12, 4, 1, 6, 3, 25, 5, 1, 58, 12, 5, 1, 222, 1], [12, 4, 1, 34, 11, 1, 9, 1], [1
3, 3, 1, 151, 6, 2, 1, 19, 7, 1, 66, 4, 1, 9, 1, 101, 43, 1, 85, 27], [20, 5, 1, 13, 18, 1, 40, 1,
34, 23, 1, 46, 1, 1, 92, 2, 1, 6, 36, 65, 1], [4, 2, 1, 78, 95, 1, 18, 51, 50, 1, 42, 1, 8, 29, 1,
18, 2, 1, 2, 33, 1], [28, 23, 1, 63, 1, 26, 54, 33, 24, 1, 5, 41, 76, 5, 1, 18, 185, 31, 1, 8, 29,
1, 22, 4], [105, 16, 1, 35, 16, 1, 115, 8, 51, 5, 1, 165, 31, 1, 3, 2, 1, 47, 13, 1, 22, 18, 1], [
13, 3, 1, 42, 1, 34, 11, 1, 66, 11, 27, 1, 34, 2, 1, 133, 116, 107, 1], [21, 77, 1, 11, 3, 1, 52,
50, 1, 27, 1, 117, 97, 74, 53, 1, 8, 29, 1, 94, 1], [59, 3, 15, 1, 48, 3, 1, 8, 29, 1, 30, 43, 1],
[37, 51, 94, 1, 86, 10, 25, 1, 28, 23, 1, 58, 16, 1, 20, 117, 57, 15, 1], [26, 7, 16, 1, 39, 1, 88
, 54, 102, 31, 1, 8, 29, 1, 67, 1], [12, 4, 1, 33, 10, 1, 80, 10, 1, 9, 1], [28, 23, 1, 63, 1, 21,
18, 27, 1, 22, 5, 1, 87, 5, 1, 49, 1, 14, 15, 10, 1, 47, 109, 65], [3, 166, 1, 16, 1, 14, 55, 1, 6
8, 1, 62, 24, 1, 69, 80, 41, 37, 1, 1], [26, 7, 1, 64, 93, 8, 1, 19, 1, 59, 3, 15, 1, 48, 3, 1, 5,
3, 62, 1, 14, 1], [5, 7, 83, 1, 8, 29, 1, 22, 1, 2, 33, 1, 4, 2, 1, 12, 4, 1, 3, 152, 1, 71], [11,
6, 16, 1, 13, 3, 1, 19, 7, 1, 8, 29, 1, 1, 3, 2, 1, 87, 1, 2, 103, 1, 14, 1, 26, 7, 1], [28, 53, 1
, 70, 38, 20, 15, 1, 186, 1, 223, 96, 1, 117, 50, 1, 14, 1, 17, 121, 1], [110, 27, 1, 224, 225, 1,
34, 2, 1, 3, 54, 56, 50, 8, 1, 49, 1, 9, 1, 50, 66, 8, 1, 78, 95, 1, 32, 10, 1, 8, 29, 1, 140, 3,
10, 1, 2, 33, 1], [78, 95, 1, 19, 7, 1, 19, 1, 8, 29, 1, 9, 1, 39, 1, 13, 18, 1], [22, 37, 1, 84,
1, 19, 108, 1, 21, 61, 81, 1, 141, 114, 2, 1, 73, 10, 1, 2, 103, 1], [6, 30, 11, 1, 42, 1, 140, 24
, 106, 37, 8, 3, 1, 21, 226, 187, 74, 1, 60, 1, 32, 90, 24, 1], [71, 1, 22, 107, 1, 35, 16, 1, 6,
227, 1], [28, 53, 1, 48, 125, 17, 1, 14, 1, 17, 43, 1, 12, 4, 1], [39, 1, 7, 188, 118, 2, 1, 9, 1]
, [28, 23, 1, 70, 38, 20, 15, 1, 1, 19, 1, 48, 125, 17, 1, 189, 10, 1, 14, 1, 2, 33, 1, 9, 1, 34,
2, 1, 34, 2, 1], [4, 2, 1, 28, 53, 1, 47, 12, 3, 11, 1, 12, 98, 1, 8, 29, 1, 5, 7, 83, 1, 2, 103],
[13, 3, 1, 35, 6, 1, 48, 3, 1, 3, 2, 119, 1], [4, 2, 1, 70, 38, 20, 15, 1, 27, 1, 134, 45, 1, 32,
24, 7, 1, 3, 2, 1, 47, 13, 1, 9, 1], [38, 37, 25, 6, 1, 37, 17, 32, 1, 60, 1, 74, 32, 1], [20, 142
, 36, 1, 18, 23, 106, 8, 1, 71, 1, 18, 32, 50, 1, 3, 77, 1, 8, 29, 1, 30, 22, 1], [12, 4, 1, 21, 7
7, 1, 26, 190, 11, 1, 14, 1, 99, 13, 55, 1], [35, 6, 1, 25, 50, 1, 27, 1, 72, 83, 16, 1, 228, 17,
1, 16, 1, 34, 11, 1, 8, 29, 1, 49, 1], [90, 106, 1, 21, 51, 38, 32, 1], [66, 11, 45, 1, 191, 6, 22
9, 2, 11, 1, 14, 15, 10, 1, 60, 1], [167, 17, 116, 2, 1, 49, 1, 64, 23, 8, 1, 89, 10, 1, 17, 43, 1
, 121, 126, 1], [12, 4, 1, 34, 11, 1, 9, 1, 4, 2, 1, 6, 153, 4, 1, 14, 15, 10, 1, 89, 10, 1, 17, 4
, 1], [25, 32, 1, 38, 37, 25, 6, 1], [121, 126, 1, 1, 1, 1, 1, 72, 79, 1, 5, 44, 127, 1, 70, 38, 2
0, 15, 1, 89, 10, 1], [39, 1, 105, 18, 1, 78, 95, 1, 230, 5, 1, 8, 29, 1, 9, 1], [92, 2, 1, 132, 4
5, 65, 1], [192, 5, 62, 1, 192, 5, 62, 1, 7, 2, 1, 8, 1, 99, 231, 1, 71], [26, 7, 1, 5, 6, 16, 1,
25, 120, 4, 31, 1, 14, 1], [35, 6, 1, 135, 24, 20, 74, 2, 1, 6, 30, 11, 1, 103, 51, 232, 17, 1, 60
, 1], [193, 33, 15, 1, 104, 1, 168, 3, 1], [71, 1, 122, 75, 1, 30, 22, 1], [12, 4, 1, 22, 37, 1, 2
33, 18, 1, 18, 154, 128, 1], [25, 32, 1, 1, 1, 1, 5, 44, 127, 1, 131, 37, 1, 6, 36, 1, 94, 1, 4, 2
], [52, 2, 67, 1, 7, 31, 31, 1, 3, 111, 1], [21, 68, 1, 194, 148, 98, 1, 25, 97, 1, 34, 195, 1, 7,
11, 1, 3, 111, 1], [110, 27, 1, 155, 1, 35, 6, 1, 48, 3, 1, 49, 1, 67, 1, 74, 32, 1, 47, 109, 1, 1
2, 98, 1, 8, 29, 1, 2, 33, 1], [63, 43, 1, 19, 7, 1, 40, 1, 8, 29, 1, 14, 1, 26, 7, 1], [13, 18, 1
, 16, 1, 78, 95, 1, 103, 51, 18, 1, 8, 29, 1, 89, 38, 1], [26, 7, 16, 1, 21, 61, 81, 1, 84, 32, 1,
71, 1, 122, 31, 1, 42, 1, 3, 2, 1, 169], [35, 143, 1, 136, 59, 4, 1, 5, 41, 76, 5, 1, 9, 1, 26, 54
, 33, 53, 1], [3, 10, 1, 49, 1, 47, 51, 97, 1, 16, 1, 170, 6, 43, 1, 49, 1, 63, 43, 1, 64, 93, 8,
1, 59, 3, 15, 1, 9, 1], [78, 95, 1, 77, 1, 8, 29, 1, 6, 63, 1, 68, 1], [22, 37, 1, 40, 1, 47, 8, 1
, 19, 108, 1, 52, 23, 1, 19, 7, 1, 14, 1, 99, 62, 1, 9, 1], [28, 23, 1, 63, 1, 13, 3, 1, 7, 31, 31
, 1, 80, 120, 13, 1, 60], [28, 24, 1, 156, 129, 5, 1, 35, 16, 1, 157, 11, 7, 1, 14, 1, 5, 3, 36, 1
, 9, 1, 87, 17, 10, 1, 1], [64, 93, 8, 1, 49, 1, 4, 142, 24, 1, 1, 19, 108, 1, 21, 61, 158, 1, 67,
1], [8, 29, 1, 86, 10, 25, 234, 55, 1, 58, 16, 1, 110, 1, 52, 50, 1, 137, 2, 1], [47, 10, 1, 122,
75, 1, 3, 2, 1, 47, 4, 1, 85, 27, 1], [4, 2, 1, 34, 2, 1, 34, 2, 1, 196, 117, 19, 1, 42, 1, 12, 98
, 1, 48, 3, 1, 14, 36, 1, 9, 1], [30, 41, 23, 1, 7, 44, 11, 1, 35, 6, 1, 89, 10, 1, 17, 121, 1], [
112, 144, 50, 5, 1, 13, 18, 1], [105, 16, 1, 42, 1, 5, 41, 76, 5, 1, 32, 28, 106, 1, 2, 103, 1, 84
, 1, 12, 4, 1, 34, 11, 1, 9, 1], [70, 38, 20, 15, 1, 46, 1, 109, 27, 1, 40, 1, 91, 13, 1, 77, 1, 7
8, 95, 1, 48, 3, 1, 3, 2, 1, 119, 1], [21, 23, 1, 25, 32, 1, 20, 11, 100, 1, 37, 51, 94, 1, 14, 1,
77, 1, 17, 4, 1], [13, 3, 1, 151, 6, 2, 1, 141, 114, 2, 1, 89, 38, 1, 9], [26, 54, 32, 24, 1, 63,
59, 1, 63, 59, 1, 104, 82, 57, 25, 31, 1, 28, 24, 1, 197, 34, 20, 10, 1, 151, 6, 2, 1, 2, 235, 44,
15, 2, 1, 3, 2, 27, 1, 40, 1, 91, 13, 1], [3, 33, 1, 7, 2, 1, 17, 13, 1], [4, 2, 1, 71, 1, 6, 30,
11, 1, 18, 23, 106, 8, 1, 14, 1, 17, 4, 1, 30, 1], [39, 1, 35, 6, 1, 25, 38, 1, 3, 33, 1, 7, 2, 1,
17, 4, 1], [35, 6, 1, 40, 1, 135, 24, 20, 74, 2, 1, 27, 1, 72, 79, 1, 236, 73, 1, 46, 1, 42, 1, 88
, 36, 2, 1, 47, 22], [87, 17, 10, 1, 6, 45, 1, 2, 109, 1, 9, 1, 20, 5, 1, 13, 18, 1, 27, 1], [68,
19, 2, 1, 13, 12, 5, 126, 69, 237, 5, 1, 60, 1, 14, 15, 10, 1, 46, 1], [12, 4, 1, 22, 18, 1, 6, 36
, 147, 107, 1, 28, 23, 1, 21, 19, 123, 15, 1, 49, 1, 13, 12, 5, 87, 57, 53, 1, 198, 15, 1], [99, 1
, 8, 29, 1, 3, 24, 1, 2, 33, 1, 110, 1, 69, 80, 41, 37, 1], [25, 32, 1, 4, 2, 1, 5, 42, 1, 117, 44
, 19, 123, 15, 1, 73, 10, 1, 2, 33, 1], [199, 140, 4, 1, 19, 1, 5, 6, 16, 1, 68, 19, 2, 1, 13, 56,
18, 1, 9, 1, 26, 54, 33, 24, 1], [64, 93, 8, 1, 27, 1, 238, 86, 1, 90, 2, 1, 3, 2, 1, 2, 200, 1, 9
, 1], [21, 23, 1, 4, 2, 1, 39, 1, 35, 6, 1, 25, 38, 1, 10, 17, 36, 1, 9, 1, 18, 2, 239, 1, 22, 13,
55, 1], [12, 4, 1, 68, 19, 2, 1, 21, 56, 18, 1, 9, 1], [72, 79, 1, 39, 1, 13, 56, 18, 1, 201, 10,
78, 10, 1, 21, 61, 81, 1, 8, 29, 1, 10, 55, 1], [21, 68, 1, 12, 4, 1, 72, 79, 1, 85, 1, 88, 51, 20
2, 1, 5, 7, 83, 1, 2, 33, 1, 9, 1], [19, 7, 1, 14, 1, 17, 4], [3, 6, 1, 11, 3, 1, 13, 3, 1, 42, 1,
34, 11, 1, 133, 116, 107, 1], [35, 6, 1, 116, 40, 75, 8, 1, 18, 2, 1, 8, 29, 1, 18, 203, 131], [28
, 53, 1, 70, 38, 20, 15, 1, 19, 1, 12, 4, 1, 44, 130, 15, 5, 1, 9, 1], [204, 23, 15, 1, 13, 18, 1]
, [6, 30, 11, 1, 42, 1, 6, 3, 25, 5, 1, 164, 6, 52, 20, 15, 1, 9, 1, 26, 54, 33, 24, 1, 101, 43, 1
], [64, 93, 8, 1, 48, 125, 17, 1, 35, 16, 1, 3, 171, 1, 5, 7, 83, 1, 46, 1, 8, 29, 1, 22, 1, 2, 33
, 1, 9, 1], [65, 57, 1, 72, 79, 1, 17, 159, 1, 5, 7, 240, 1, 9, 1, 12, 4, 1], [72, 79, 1, 84, 1, 3
9, 1, 13, 56, 18, 1, 6, 30, 11, 1, 21, 61, 81, 1, 10, 17, 36, 1, 9, 1], [21, 23, 1, 25, 32, 1, 12,
4, 1, 34, 11, 1, 9, 1], [28, 53, 1, 66, 44, 7, 11, 1, 20, 11, 100, 1, 241, 24, 6, 1, 35, 16, 1, 9,
1], [35, 6, 1, 25, 50, 1, 27, 1, 72, 83, 16, 1, 6, 31, 11, 102, 134, 1, 49, 1], [70, 38, 20, 15, 1
, 46, 1, 6, 30, 11, 1, 7, 172, 1, 22, 4], [3, 10, 1, 19, 1, 64, 93, 8, 1, 21, 164, 50, 6, 10, 1, 9
, 1], [4, 2, 1, 12, 4, 1, 136, 59, 4, 1, 5, 41, 76, 5, 1, 9, 1], [242, 243, 1, 92, 2, 1, 5, 44, 36

, 1, 70, 38, 20, 15, 1, 48, 3, 1, 14, 1, 5, 3, 36, 1, 60, 1], [3, 6, 1, 11, 3, 1, 73, 129, 107, 1, 1, 20, 11, 27, 1, 58, 16, 1, 46, 1], [6, 30, 11, 1, 21, 61, 81, 1, 101, 43, 1], [58, 16, 1, 86, 10, 25, 27, 1, 40, 1, 91, 13, 1, 104, 82, 57, 25, 31, 1, 28, 23, 1, 4, 2, 1], [22, 1, 33, 1, 1, 1, 1, 12, 4, 1, 34, 11, 1, 9, 1], [13, 3, 31, 7, 1, 68, 19, 2, 1], [28, 53, 1, 64, 93, 8, 1, 96, 20, 7, 1, 16, 1, 73, 10, 1, 2, 42, 1, 9, 1, 45, 1], [32, 46, 1, 22, 18, 16, 1, 39, 1, 88, 54, 102, 31, 1, 8, 29, 1, 67], [39, 1, 13, 56, 18, 1, 3, 77, 1, 77, 1, 244, 109, 1, 113, 1, 5, 3, 127, 1, 9, 1], [22, 245, 73, 41, 57, 73, 66, 11, 1, 3, 2, 1, 47, 4, 1, 26, 7, 27, 1], [4, 2, 1, 28, 23, 1, 58, 16, 1, 25, 18, 5, 1, 3, 6, 1, 89, 129, 107, 1], [28, 53, 1, 35, 6, 1, 12, 98, 1, 8, 29, 1, 22, 1, 2, 42, 1, 9, 1], [205, 53, 1, 4, 2, 1, 17, 10, 127, 1, 14, 1, 17, 121, 1], [206, 53, 1, 173, 86, 1], [4, 2, 1, 39, 1, 84, 1, 7, 246, 124, 8, 1, 247, 52, 1, 19, 7, 1, 3, 2, 36, 1, 9, 1], [62, 23, 1, 63, 1, 13, 3, 1, 34, 2, 1, 46, 1, 5, 7, 83, 1, 46, 1, 8, 29, 1, 22, 127, 1, 12, 4, 1], [174, 3, 112, 1, 52, 2, 67, 1, 22, 57, 1, 10, 74, 1, 112, 1, 1], [6, 30, 11, 1, 18, 23, 106, 8, 1, 3, 2, 36, 1, 9, 1, 39, 1, 13, 18, 1], [136, 59, 4, 1, 5, 41, 76, 5, 1], [47, 25, 97, 1, 46, 1, 70, 38, 20, 64, 5, 1, 49, 1, 23, 15, 1, 19, 108, 1, 28, 207, 128, 1, 14, 1, 99, 127, 1, 9, 1], [22, 37, 1, 19, 1, 138, 5, 7, 1, 35, 52, 1, 9, 1], [104, 82, 57, 25, 31, 1, 101, 43, 1], [7, 145, 1, 22, 1, 17, 4, 1, 101, 43, 1], [64, 93, 8, 1, 59, 3, 15, 1, 48, 3, 1, 3, 2, 1, 5, 3, 36, 1, 9, 1, 85, 1], [26, 7, 1, 6, 30, 11, 1, 21, 61, 248, 1, 14, 1], [20, 8, 40, 1, 143, 140, 57, 2, 1, 13, 204, 134, 12, 112, 59, 74, 1, 16, 1, 3, 54, 56, 50, 8, 1, 3, 2, 100, 1, 18, 154, 128, 1], [21, 23, 1, 133, 116, 1, 101, 43, 1], [39, 1, 12, 4, 1, 9, 1], [12, 4, 1, 26, 7, 1, 28, 53, 1, 103, 51, 18, 1, 3, 2, 1, 5, 3, 62, 1, 14, 1], [13, 18, 1, 28, 1, 31, 7, 1, 8, 29, 1, 9, 1], [26, 7, 16, 1, 39, 1, 88, 54, 102, 31, 1, 8, 29, 1, 67, 1], [72, 79, 1, 26, 7, 16, 1, 39, 1, 88, 54, 102, 31, 1, 8, 29, 1, 67, 1], [21, 68, 1, 175, 5, 160, 40, 1], [208, 86, 1, 141, 114, 1, 3, 2, 1, 47, 4, 1], [90, 106, 1, 21, 51, 38, 32, 1, 12, 4, 1, 19, 7, 1, 66, 4, 1, 9, 1, 26, 7, 27, 1], [7, 44, 11, 1, 14, 15, 10, 1, 60, 1, 4, 2, 1], [25, 65, 1, 1, 1, 1], [26, 7, 63, 1, 78, 95, 1, 5, 7, 83, 1, 46, 1, 8, 29, 1, 22, 36, 1, 12, 4, 1, 6, 32, 8, 137, 31, 1, 1], [35, 6, 1, 135, 24, 20, 74, 2, 1, 17, 159, 1, 60, 1], [68, 32, 11, 53, 8, 1], [173, 86, 1, 5, 41, 76, 5, 1], [20, 11, 27, 1, 19, 7, 1, 58, 16, 1, 46, 1, 20, 11, 45, 1, 115, 8, 51, 5, 1], [13, 3, 1, 34, 2, 1, 46, 1, 5, 7, 83, 1, 8, 29, 1, 22, 4, 1, 133, 38, 1, 69, 80, 41, 37, 1, 3, 2, 27, 1, 84, 1], [65, 13, 1, 14, 13, 1, 1, 1, 28, 24, 1, 64, 93, 8, 1, 59, 3, 15, 1, 48, 3, 1, 14, 1, 17, 4, 1], [122, 75, 1, 30, 22, 1, 110, 1], [32, 28, 106, 1, 105, 16, 1, 35, 16, 1, 14, 1, 99, 36, 1, 9, 1, 68, 74, 176, 114, 1, 13, 18], [14, 15, 10, 1, 177, 8, 1, 16, 1, 20, 38, 40, 1, 46, 1, 9, 1], [135, 24, 20, 74, 2, 1, 27, 1, 178, 161, 10, 1, 3, 2, 1, 47, 4, 1], [21, 23, 1, 39, 1, 115, 8, 10, 1, 35, 16, 1, 249, 3, 1, 14, 1, 17, 4, 1, 17, 179, 1], [22, 37, 1, 19, 1, 36, 18, 90, 8, 1, 12, 4, 1, 9, 1], [21, 23, 1, 4, 2, 1, 3, 44, 15, 7, 2, 1, 40, 57, 2, 1, 63, 1, 250, 10, 1, 10, 55, 62, 1, 10, 55, 62, 1, 18, 23, 106, 8, 1, 14, 1, 17, 4, 1, 30], [68, 19, 2, 1, 35, 6, 1, 67, 1, 4, 2], [26, 7, 1, 13, 3, 1, 31, 7, 1, 68, 74, 251, 252, 1, 14, 1], [71, 1, 26, 7, 16, 1, 45, 122, 75, 1, 162, 1], [66, 11, 45, 1, 5, 42, 1, 21, 56, 18, 1, 9, 1, 4, 2, 1], [28, 23, 1, 21, 19, 123, 15, 1, 46, 1, 66, 11, 45, 1, 156, 129, 5, 1, 9, 1], [35, 6, 1, 3, 6, 1, 11, 3, 1, 22, 13, 55, 1], [19, 75, 1, 39, 1, 18, 32, 50, 1, 14, 1, 99, 36, 1], [174, 12, 5, 1, 52, 2, 253], [39, 1, 13, 56, 18, 1, 6, 30, 11, 1, 114, 5, 27, 1, 10, 17, 1, 17, 4, 1, 9, 1, 4, 2, 1], [6, 30, 11, 1, 7, 145, 1, 22, 4, 1], [26, 7, 1, 6, 154, 1, 7, 188, 118, 2, 1, 14, 1], [26, 7, 16, 1, 21, 61, 81, 1, 17, 179, 1, 16, 1, 19, 7, 1, 3, 24, 1, 169], [39, 1, 13, 56, 18, 1, 20, 44, 62, 90, 10, 1, 3, 2, 40, 1, 113, 254, 1], [105, 143, 1, 5, 41, 76, 5, 1, 49, 1, 88, 54, 102, 31, 1, 67, 1, 22, 18, 16, 1], [22, 57, 1, 13, 7, 1, 20, 54, 56, 23, 44, 5, 86, 1, 1], [17, 82, 118, 1, 14, 15, 10, 1, 60, 1], [22, 124, 53, 1, 36, 53, 124, 1, 12, 4, 1, 9, 1, 156, 129, 5, 1, 13, 12, 5, 87, 57, 2, 1, 14, 27, 1, 46, 1], [14, 15, 10, 1, 19, 1, 109, 45, 1, 25, 120, 4, 11, 1, 60, 1], [32, 24, 7, 145, 113, 1], [110, 1, 201, 10, 78, 10, 1, 185, 26, 255, 15, 1, 146, 1, 20, 8, 40, 1, 5, 41, 76, 5, 1, 16, 1], [12, 4, 1, 180, 10, 85, 1, 196, 15, 2, 256, 5, 1, 9, 1, 26, 54, 33, 23, 1, 13, 56, 18, 1, 19, 1], [22, 18, 1, 21, 18, 100, 1, 5, 41, 76, 5, 1, 21, 18, 27, 1, 87, 5, 1, 2, 209, 1], [58, 16, 1, 86, 10, 25, 27, 1, 40, 1, 91, 13, 1, 163, 12, 69, 4, 1, 4, 24, 1], [12, 4, 1, 110, 1, 115, 8, 51, 5, 1, 105, 86, 1, 3, 2, 1, 5, 3, 36, 1, 146, 1], [47, 10, 1, 134, 11, 1, 91, 4, 1, 85, 27, 1], [183, 3, 1, 257, 3, 1, 13, 56, 18, 1, 10, 17, 36, 1, 9, 1], [71, 1, 84, 1, 6, 30, 11, 1, 122, 75, 1, 146, 1, 39, 1, 13, 56, 18, 1, 112, 5, 1, 3, 2, 40, 1], [28, 32, 2, 34, 100, 1, 22, 18, 49], [21, 18, 45, 1, 58, 3, 1, 21, 18, 27, 1, 87, 5, 1, 2, 124, 1], [6, 30, 11, 1, 17, 41, 102, 1, 9, 1, 4, 2, 1, 57, 166, 1], [47, 90, 17, 1, 7, 11, 1, 157, 24, 6, 1, 3, 2, 1, 92, 2, 1, 37, 51, 94, 1, 69, 80, 41, 37, 1, 3, 2, 1], [21, 23, 1, 25, 32, 1], [39, 1, 35, 52, 1, 136, 59, 4, 1, 56, 38, 8, 1, 9, 1], [68, 149, 44, 11, 1, 135, 24, 20, 74, 2, 1, 60, 1], [57, 32, 1, 84, 1, 6, 30, 11, 1, 21, 61, 81, 1, 105, 18, 1, 9], [12, 4, 1, 72, 79, 1, 38, 44, 15, 1, 198, 15, 1, 6, 36, 147, 107, 1], [21, 77, 1, 40, 1, 21, 77, 1, 48, 3, 1, 3, 111, 1], [21, 17, 50, 1, 7, 42, 27, 1, 49, 1, 70, 38, 20, 15, 1, 49, 1, 59, 3, 15, 1, 3, 24, 1, 119, 1, 258, 17, 259, 260, 1, 16, 1, 47, 51, 97, 1, 49], [4, 2, 1, 39, 1, 6, 30, 11, 1, 5, 42, 1, 131, 144, 1, 9, 1], [21, 6, 1, 12, 4, 1, 3, 152, 1], [19, 75, 1, 39, 1, 14, 1, 87, 36, 1, 4, 2, 1], [52, 50, 1, 28, 24, 1, 58, 52, 1, 3, 6, 1, 11, 3, 1, 22, 13, 55, 1], [85, 1, 17, 159, 1, 9, 1, 12, 4, 1], [5, 6, 16, 1, 5, 44, 36, 1, 70, 38, 20, 15, 1, 19, 1, 59, 3, 15, 1, 48, 3, 1, 3, 2, 1, 119, 1, 20, 104, 2, 1, 16, 1, 170, 6, 43, 1, 19], [64, 23, 8, 1, 12, 4, 1, 96, 20, 7, 1, 44, 130, 75, 8, 1, 18, 203, 181, 128], [13, 3, 1, 88, 6, 2, 1, 48, 3, 1, 3, 111, 1], [18, 23, 106, 8, 1, 14, 1, 17, 4, 1, 261, 1, 4, 2], [104, 82, 57, 25, 31, 1, 110, 1, 122, 75, 1, 30, 22, 1], [92, 2, 1, 12, 4, 1, 12, 4, 1, 3, 2, 36, 1, 9, 1, 39, 1, 13, 18], [7, 43, 1, 6, 30, 11, 1, 126, 262, 11, 1, 30], [14, 15, 10, 1, 16, 1, 193, 112, 1, 21, 61, 81, 1, 8, 29, 1, 60, 1], [20, 8, 19, 1, 5, 41, 76, 5, 1, 5, 7, 83, 1, 40, 1, 34, 32, 2, 1, 9, 1], [72, 79, 1, 70, 38, 20, 15, 1, 19, 1, 59, 3, 15, 1, 89, 10, 1, 17, 4, 1, 25, 32, 1], [26, 7, 1, 92, 2, 1, 68, 15, 2, 1, 3, 2, 1, 5, 3, 62, 1, 14, 1], [28, 24, 1, 47, 90, 17, 1, 157, 24, 6, 1, 7, 11, 1, 3, 2, 1, 92, 2, 1, 26, 190, 11, 1, 69, 80, 41, 37, 1, 3, 2, 1, 1, 1], [6, 160, 1, 55, 160, 1, 48, 3, 1, 3, 2, 1, 5, 3, 62, 1, 14, 1, 12, 4, 1], [22, 37, 1, 19, 1, 138, 5, 7, 1, 6, 30, 11, 1, 6, 153, 4, 1, 9, 1, 4, 2, 1], [210, 148, 4, 87, 1, 165, 104, 1, 3, 111, 1, 26, 54, 33, 24, 1], [26, 7, 1, 66, 143, 1, 19, 7, 1, 40, 1, 8, 29, 1, 14, 1, 6, 32, 8, 137, 31, 1], [12, 4, 1, 6, 36, 65, 1, 4, 2, 1, 208, 86, 1, 42, 1, 157, 24, 6, 1, 9, 1], [34, 32, 2, 1, 12, 4, 1, 263, 18, 23, 73, 2, 1, 9], [3, 77, 1, 84, 1, 19, 7, 1, 40, 1, 22, 65, 1, 26, 7, 1], [20, 11, 100, 1, 264, 31, 128, 1, 67, 1, 74, 32, 1, 18, 2, 1], [65, 13, 1, 14, 13, 1, 1, 1, 1], [106, 34, 75, 1], [174, 12, 112, 1, 206, 53

, 1, 7, 73, 1], [7, 145, 1, 8, 29, 1, 22, 4, 1, 20, 5, 1, 34, 2], [21, 23, 1, 6, 153, 4, 1, 13, 3, 1, 103, 51, 18, 1, 3, 2, 1, 119, 1, 52, 2, 67], [28, 24, 1, 70, 38, 20, 15, 1, 22, 265, 2, 19, 2, 1, 48, 3, 1, 14, 1, 17, 4, 1, 30, 41, 23, 1, 1, 1], [6, 30, 11, 1, 42, 1, 136, 59, 4, 1, 13, 56, 18, 1, 9, 1, 26, 54, 33, 24, 1], [71, 1, 19, 108, 1, 266, 3, 1, 17, 4, 1, 22, 37, 1], [12, 4, 1, 63, 43, 1, 176, 18, 8, 1, 9, 1, 26, 54, 33, 23, 1, 87, 5, 1], [4, 2, 1, 12, 4, 1, 6, 3, 25, 5, 1, 3, 2, 1, 2, 103, 1, 14, 1, 26, 7, 1], [35, 6, 1, 22, 27, 1, 46, 1, 92, 2, 1, 66, 11, 45, 1, 96, 20, 7, 1, 9, 1], [26, 7, 27, 1, 7, 8, 1, 56, 2, 267, 51, 91, 11, 1, 3, 2, 1, 47, 4, 1], [4, 2, 1, 3, 42, 1, 268, 7, 27, 1, 73, 10, 62, 1, 9, 1], [12, 4, 1, 25, 120, 4, 11, 1, 9, 1, 4, 2, 1], [20, 11, 45, 1, 28, 207, 55, 1, 12, 98, 1, 175, 44, 117, 40, 1], [78, 95, 1, 58, 16, 1, 9, 1, 4, 1, 8, 29, 1, 28, 23, 1, 21, 19, 123, 15, 1, 46, 1], [12, 4, 1, 5, 42, 1, 13, 18, 1, 9, 1, 39, 1], [17, 25, 2, 1, 85, 1, 37, 51, 94, 1, 69, 80, 41, 37, 1, 3, 23, 55, 1], [3, 102, 27, 1], [21, 61, 81, 1, 19, 7, 1, 66, 4, 1, 101, 43, 1], [6, 30, 11, 1, 191, 6, 1], [17, 13, 1, 173, 37, 23, 1, 14, 1, 26, 7, 1, 32, 24, 7, 99, 113], [211, 162, 1, 1, 1, 7, 145, 1, 22, 1, 17, 4, 1], [26, 54, 33, 53, 1, 5, 41, 76, 5, 1, 16, 1, 32, 7, 1, 122, 75, 1, 9], [105, 52, 1, 10, 17, 1, 2, 33, 1, 9, 1, 94, 25, 2, 1, 16, 1, 34, 11, 1, 3, 2, 1, 2, 33, 1, 30, 1], [20, 11, 27, 1, 58, 16, 1, 46, 1, 66, 11, 27, 1, 47, 8, 1, 73, 129, 107, 1], [72, 79, 1, 26, 7, 16, 1, 34, 11, 1, 8, 29, 1, 3, 2, 100, 1, 9, 1], [22, 1, 33, 1, 52, 2, 67, 1, 26, 7, 1, 38, 37, 25, 6, 1, 14, 1], [12, 4, 1, 28, 24, 1, 70, 38, 20, 15, 1, 141, 8, 1, 96, 20, 7, 1, 9, 1], [112, 1, 21, 23, 1, 269, 1, 44, 212, 15, 1], [22, 37, 1, 19, 1, 138, 5, 7, 1, 6, 36, 65, 1], [6, 30, 11, 1, 199, 200, 1, 146, 1, 71, 1, 26, 54, 33, 23, 1, 19, 2, 213], [52, 50, 1, 90, 31, 2, 137, 31, 1, 14, 1, 26, 7, 1], [149, 15, 1, 4, 2, 1, 28, 53, 1, 70, 38, 20, 15, 1, 117, 50, 1, 14, 1, 17, 121, 1], [28, 24, 1, 58, 3, 1, 3, 6, 1, 11, 3, 1, 73, 50, 55], [210, 18, 1, 80, 18, 1, 28, 23, 1, 58, 16, 1, 270, 96, 1], [17, 25, 2, 1, 13, 18, 1], [13, 3, 1, 19, 7, 1, 3, 2, 1, 119], [25, 65, 1, 167, 17, 116, 2, 1, 49, 1, 64, 23, 8, 1, 89, 10, 1, 17, 43], [22, 39, 1, 14, 39, 1, 1, 1, 1, 22, 37, 1, 84, 1, 85, 27, 1, 47, 10, 1, 134, 11, 1, 91, 4, 1, 101, 43, 1], [5, 6, 16, 1, 5, 44, 127, 1, 70, 38, 20, 15, 1, 35, 16, 1, 18, 36, 1, 73, 50, 128, 1], [6, 30, 11, 1, 42, 1, 88, 82, 118, 1], [39, 1, 35, 6, 1, 150, 2, 16, 1, 16, 1, 96, 20, 7, 1, 139, 1, 48, 3, 1, 8, 29, 1, 30, 43, 1], [6, 32, 8, 137, 31, 1, 1], [138, 5, 7, 1, 12, 4, 1, 132, 33, 45, 1, 9, 1], [121, 126, 1, 1, 1, 1, 71, 1, 37, 51, 94, 1, 136, 2, 1, 22, 1, 17, 4, 1], [5, 6, 16, 1, 3, 7, 1, 25, 97, 1, 68, 5, 1, 58, 3, 1, 177, 8, 1, 143, 1, 9, 1, 1], [28, 53, 1, 70, 38, 20, 15, 1, 50, 15, 1, 12, 98, 1, 30, 43, 1, 9, 1], [20, 104, 2, 1, 16, 1, 48, 125, 17, 1, 12, 98, 1, 8, 29, 1, 14, 1, 2, 33, 1, 9, 1, 1], [14, 15, 10, 1, 49, 1, 116, 40, 75, 8, 1, 21, 61, 158, 1, 8, 42, 1, 67, 1], [90, 31, 2, 137, 31, 1], [20, 5, 40, 1, 57, 166, 1, 16, 1, 21, 6, 1, 5, 41, 76, 5, 1, 8, 29, 1, 169, 55, 1], [6, 30, 11, 1, 42, 1, 25, 120, 4, 31, 1, 5, 41, 76, 5, 1, 9, 1, 14, 15, 10, 1, 49, 1, 1], [6, 271, 11, 102, 214, 1, 44, 96, 114, 1, 60, 1], [72, 79, 1, 16, 114, 1, 108, 10, 1, 8, 29, 1, 14, 1, 2, 33, 1, 60, 1], [66, 11, 100, 1, 34, 2, 1, 13, 3, 1, 42, 1, 131, 144, 1, 133, 10, 100, 1, 18, 154, 128, 1], [6, 30, 11, 1, 96, 20, 7, 1, 10, 17, 1, 2, 33, 1, 9, 1, 48, 3, 1, 32, 27, 1, 28, 1], [177, 8, 1, 52, 1, 56, 38, 8, 1, 18, 69, 25, 2, 1, 40, 1, 120, 52, 6, 1, 16, 1, 5, 42, 1, 14, 55, 1, 1], [71, 1, 215, 59, 44, 180, 20, 86, 1, 8, 29, 1, 30, 1], [19, 108, 1, 7, 31, 31, 1, 3, 2, 1, 94, 1, 85, 27], [25, 65, 1, 1, 1, 1, 4, 2, 1, 12, 4, 1, 131, 214, 1, 6, 36, 43, 1, 9, 1, 26, 7, 27], [78, 95, 1, 7, 172, 1, 8, 42, 1, 22, 4, 1], [112, 144, 50, 5, 1], [28, 23, 1, 64, 93, 8, 1, 19, 1, 48, 125, 17, 1, 189, 10, 1, 14, 1, 2, 33, 1, 9, 1, 34, 2, 1, 34, 2, 1], [21, 18, 27, 1, 22, 18, 1, 63, 1, 5, 7, 83, 1, 12, 4, 1, 2, 109, 1, 9, 1, 85, 27, 1, 78, 142, 62, 1], [105, 52, 1, 35, 16, 1, 73, 50, 55, 1], [21, 134, 6, 1, 87, 17, 10, 1, 14, 1, 26, 7, 1, 68, 1], [12, 4, 1, 155, 2, 118, 2, 1, 19, 7, 1, 66, 4, 1, 9, 1, 101, 43], [5, 41, 76, 5, 1, 20, 54, 56, 171, 74, 1, 3, 111, 1], [22, 43, 1, 272, 5, 1, 13, 12, 5, 139, 12, 182, 17, 1, 197, 2], [66, 11, 45, 1, 5, 273, 1, 14, 15, 10, 1, 60, 1], [85, 1, 6, 5, 1, 21, 18, 45, 1, 58, 12, 5, 1, 117, 130, 51, 5, 1, 113, 1, 45, 38, 57, 3, 1], [103, 51, 18, 1, 3, 2, 27, 1, 40, 1, 91, 13, 1, 104, 82, 57, 25, 31, 1], [57, 32, 1, 14, 15, 10, 1, 19, 108, 1, 21, 61, 81, 1, 60, 1], [168, 3, 1, 141, 168, 1], [32, 7, 53, 1, 4, 142, 24, 1, 132, 124, 31, 1, 8, 29, 1, 67, 1], [112, 1, 22, 2, 1, 31, 1, 68, 44, 15], [28, 23, 1, 21, 19, 123, 15, 1, 19, 1, 156, 129, 5, 1, 6, 36, 39, 1, 37, 24], [19, 108, 1, 274, 275, 86, 1, 9, 1], [47, 90, 17, 1, 124, 24, 6, 1, 3, 2, 1, 47, 4, 1], [5, 42, 1, 9, 1], [22, 37, 19, 1, 138, 5, 7, 1, 17, 82, 118, 1, 9, 1], [21, 23, 1, 25, 32, 1, 12, 4, 1, 5, 41, 76, 5, 1, 9, 1], [7, 44, 11, 1, 34, 69, 75, 1, 30, 43, 1, 22, 37, 1, 25, 32, 1], [33, 10, 1, 150, 10, 33, 10, 1, 46, 1, 34, 69, 75, 1, 14, 27, 1, 25, 97, 1, 9, 1, 12, 4], [101, 43, 1, 72, 79, 1, 13, 3, 1, 34, 11, 1, 6, 36, 65, 1], [58, 16, 1, 21, 77, 1, 11, 3, 1, 25, 18, 5, 1, 8, 29, 1, 22, 13, 1, 4, 2, 1], [63, 161, 5, 1, 3, 2, 100, 1, 80, 120, 13, 1], [35, 52, 1, 9, 1, 85], [71, 1, 139, 46, 15, 1, 35, 16, 1, 3, 171, 1], [26, 54, 33, 24, 1, 63, 43, 1, 37, 25, 6, 1, 8, 29, 1, 52, 2, 67], [28, 24, 1, 13, 3, 1, 5, 25, 10, 1, 9, 1, 1], [28, 23, 1, 21, 19, 123, 15, 1, 46, 1, 66, 11, 45, 1, 24, 149, 1, 6, 80, 1, 9, 1], [28, 24, 1, 47, 90, 17, 1, 124, 24, 6, 1, 14, 1, 17, 4, 1, 9, 1, 20, 16, 1, 24, 44, 36, 1, 6, 36, 62, 1], [1, 26, 79, 1, 13, 3, 1, 34, 2, 1, 110, 1, 5, 7, 83, 1, 8, 29, 1, 22, 36, 1, 12, 4, 1], [14, 15, 10, 1, 84, 1, 38, 37, 25, 6, 1, 9, 1], [155, 1, 115, 8, 10, 1, 71, 1, 8, 29, 1, 113, 124, 36, 1, 88, 5, 40, 1, 58, 16, 1, 12, 98, 1, 276, 1], [32, 7, 1, 5, 42, 1, 24, 44, 62, 1, 99, 1, 2, 103, 1, 277, 1], [21, 61, 81, 1, 13, 12, 5, 126, 69, 147, 5, 1, 8, 29, 1, 60, 1], [12, 4, 1, 6, 11, 102, 278, 1, 9, 1, 4, 2, 1], [39, 1, 115, 8, 51, 5, 1, 19, 1, 23, 15, 1, 20, 11, 45, 1, 279, 4, 118, 1, 12, 98, 1, 3, 2, 1, 113, 62, 1, 9, 1, 34, 2, 1, 34, 2, 1], [20, 8, 1, 115, 8, 51, 5, 1, 49, 1, 12, 280, 69, 216, 1, 5, 73, 1, 46, 1, 6, 30, 11, 1, 21, 61, 158, 1, 9, 1], [17, 144, 6, 1, 19, 1, 14, 15, 10, 1, 60, 1], [30, 41, 23, 1, 1, 1, 1, 13, 3, 1, 151, 6, 2, 1, 176, 18, 8, 1, 89, 10, 1, 17, 4, 1], [78, 95, 1, 105, 52, 1, 217, 10, 1, 119, 1, 4, 2, 1, 49, 1, 7, 99, 1, 1, 99, 13, 1], [13, 18, 1, 46, 1, 92, 2, 1, 132, 159, 2, 1, 38, 65, 1], [12, 4, 1, 26, 7, 27, 1, 109, 45, 1, 109, 1, 91, 4, 1], [211, 162, 1, 1], [28, 24, 1, 35, 6, 1, 178, 161, 10, 1, 3, 2, 1, 119, 1], [12, 4, 1, 68, 19, 2, 1, 49, 1, 34, 195, 1, 3, 2, 1, 2, 103, 1, 14, 1], [21, 68, 1, 13, 3, 1, 65, 281, 1, 48, 3, 1, 3, 2, 1, 119, 1, 101, 43, 1, 40, 1, 91, 13, 1], [26, 7, 1, 3, 77, 1, 78, 95, 1, 5, 7, 83, 62, 1, 42, 1, 8, 29, 1, 14, 1], [35, 6, 1, 135, 24, 20, 74, 2, 1, 27, 1, 72, 79, 1, 55, 97, 1, 94, 1], [13, 3, 1, 6, 218, 4, 1, 52, 1, 14, 15, 10, 1, 48, 3, 1, 3, 111, 1], [25, 65, 1, 1, 1, 28, 53, 1, 64, 23, 8, 1, 37, 51, 94, 1, 22], [1, 56, 97, 37, 1, 28, 23, 1, 91, 13, 1, 167, 17, 116, 2, 1, 46, 1, 13, 3, 1, 14, 130, 10, 1, 209, 155, 1], [39, 1, 14, 15, 10, 1, 19, 1, 56, 24, 20, 5, 1, 20, 11, 45, 1, 282, 4, 118, 1, 12, 98, 1, 47, 109, 1, 2, 33, 1, 9, 11, [30, 41, 23, 1, 1,

```
1, 1, 12, 4, 1, 217, 10, 1, 89, 97, 1, 9, 1], [71, 1, 26, 7, 16, 1, 6, 30, 11, 1, 12, 111, 283, 11
, 1, 30, 1], [48, 125, 17, 1, 35, 16, 1, 14, 55, 1], [32, 24, 102, 1], [6, 30, 11, 1, 42, 1, 6, 21
8, 4, 1], [26, 7, 27, 1, 72, 79, 1, 6, 30, 11, 1, 140, 24, 75, 1, 3, 2, 1, 47, 4], [39, 1, 115, 8,
10, 1, 35, 16, 1, 80, 202, 1, 3, 152, 1], [73, 116, 1, 57, 32, 1, 3, 2, 40, 1, 113, 284], [28, 53,
1, 4, 142, 24, 1, 78, 75, 10, 1, 285, 17, 10, 1, 67, 1, 104, 82, 57, 25, 31, 1, 1], [45, 20, 5, 1]
, [26, 7, 16, 1, 63, 43, 1, 19, 7, 1, 8, 1, 14, 36, 1, 175, 5, 160, 40, 1], [4, 2, 1, 3, 6, 1, 14,
55, 1, 69, 80, 41, 37, 1, 28, 23, 1, 21, 19, 123, 15, 1, 19, 1], [14, 15, 10, 1, 28, 53, 1, 13, 12
, 5, 139, 12, 130, 219, 5, 1, 40, 1, 120, 52, 6, 1, 16, 1, 8, 29, 1, 60, 1], [65, 13, 1, 13, 3, 1,
48, 125, 17, 1, 3, 2, 1, 113, 1, 35, 6, 1, 49], [28, 53, 1, 64, 93, 8, 1, 19, 1, 44, 130, 15, 5, 1
, 18, 36, 1, 3, 111, 1], [28, 24, 1, 21, 19, 123, 15, 1, 3, 6, 1, 11, 3, 1, 73, 50, 55, 1], [20, 5
, 63, 1, 77, 1, 13, 3, 1, 34, 2, 1, 181, 3, 1, 3, 2, 45, 1, 80, 120, 13, 1], [20, 104, 2, 1, 63, 4
3, 1, 286, 1, 44, 96, 2, 1, 14, 15, 10, 1, 9, 1, 22, 5, 1, 87, 5, 1, 71, 1], [39, 1, 84, 1, 21, 61
, 81, 1, 9, 1], [35, 6, 1, 25, 50, 1, 27, 1, 150, 2, 1, 16, 1, 178, 161, 10, 1, 3, 2, 1, 47, 4], [
21, 23, 1, 39, 1, 35, 16, 1, 14, 55, 1, 4, 2, 1], [26, 54, 33, 23, 1, 19, 7, 1, 49, 1, 118, 31, 1,
113, 36, 1, 146], [53, 44, 96, 41, 15, 1, 3, 2, 27, 1, 139, 1, 77, 1, 8, 29, 1, 14, 1, 2, 33, 1],
[37, 51, 94, 1, 13, 3, 1, 64, 93, 8, 1, 19, 1, 59, 3, 15, 1, 48, 3, 1, 3, 111, 1], [12, 4, 1, 3, 7
, 10, 1, 3, 2, 1, 47, 4, 1, 85, 27, 1], [35, 6, 1, 25, 50, 1, 27, 1, 72, 83, 16, 1, 287, 10, 1, 28
8, 12, 5, 1, 187, 2, 25, 4, 1], [22, 18, 16, 1, 34, 11, 1, 3, 2, 40, 1, 21, 61, 81, 1, 10, 55, 1],
[66, 11, 100, 1, 34, 2, 1, 133, 38, 1, 49, 1, 28, 24, 1, 117, 75, 1, 216, 212, 1, 69, 80, 41, 37,
1, 3, 2, 45, 1, 9, 1], [6, 30, 11, 1, 6, 153, 4, 1, 30, 22, 1, 66, 1, 58, 16, 1, 5, 7, 57, 1, 16,
1, 18, 32, 50, 1, 22, 1, 17, 13, 1], [71, 1, 21, 8, 215, 59, 44, 180, 20, 86, 1, 162, 1], [22, 18,
1, 105, 16, 1, 35, 16, 1, 28, 24, 1, 58, 52, 1, 19, 15, 1, 5, 3, 62, 1, 14, 1], [20, 5, 1, 138, 5,
7, 1, 40, 1, 19, 2, 213, 1, 71, 1, 126, 289, 11, 1, 30, 1], [26, 7, 16, 1, 8, 29, 1, 14, 1, 87, 39
, 55, 1], [7, 8, 1, 56, 2, 5, 82, 8, 1, 14, 1, 17, 4, 1, 28, 24, 1], [72, 79, 1, 62, 23, 1, 16, 1,
92, 2, 1, 88, 54, 102, 31, 1, 67, 1, 4, 2, 1], [45, 17, 290, 2, 1, 16, 1, 170, 6, 43, 1, 49, 1, 18
6, 1, 70, 38, 20, 15, 1, 59, 3, 64, 5, 1, 48, 3, 1, 3, 2, 45], [72, 79, 1, 26, 7, 16, 1, 92, 2, 1,
13, 12, 5, 139, 12, 130, 219, 5, 1, 67, 1], [21, 61, 158, 1, 35, 6, 1, 89, 10, 1, 17, 121, 1, 1, 2
5, 32, 1, 1], [78, 95, 1, 77, 1, 3, 2, 40, 1, 28, 23, 1, 58, 16, 1, 69, 74, 41, 15, 1, 3, 24, 65,
1], [92, 2, 1, 12, 4, 1, 12, 4, 1, 3, 2, 1, 5, 3, 62, 1, 14, 1, 26, 7, 1, 291, 23, 1, 91, 13, 1],
[17, 179, 1, 40, 1, 6, 61, 181, 1, 6, 36, 1, 39, 1, 56, 292, 293, 10, 7, 1, 35, 16, 1, 205, 51, 74
, 1, 3, 152, 1, 110, 1], [88, 5, 1, 14, 15, 10, 1, 46, 1, 7, 172, 1, 8, 29, 1, 22, 4], [12, 4, 1,
4, 2, 1, 21, 6, 1, 11, 3, 1, 58, 52, 1, 8, 29, 1, 22, 4, 1], [132, 82, 31, 2, 1, 13, 18, 1, 9, 1],
[104, 82, 57, 25, 31, 1, 28, 23, 1, 101, 43, 1], [194, 148, 4, 1, 9, 1, 12, 4, 1, 294, 1]]
[[ 28  23   1 ...   0   0   0]
 [163  12  69 ...   0   0   0]
 [ 39   1  14 ...   0   0   0]
 ...
 [132  82  31 ...   0   0   0]
 [104  82  57 ...   0   0   0]
 [194 148   4 ...   0   0   0]]
```

Transfrom Test Char-tokenizer using from 400 words to last and Using Train Tokenizer

In [28]:

```
from keras.preprocessing.text import Tokenizer

# integer encode the documents
encoded_docs_test_char = t_char.texts_to_sequences(char_sentence[400:])
print(encoded_docs_test_char)
# pad documents to a max length of 4 words
max_length = 45
padded_docs_test_char = sequence.pad_sequences(encoded_docs_test_char, maxlen=max_length, padding='
post')
print(padded_docs_test_char)
```

```
[[26, 7, 1, 6, 153, 4, 1, 14, 1, 4, 2, 1], [174, 12, 5, 1], [121, 126, 1, 1, 1, 1, 28, 24, 1, 70,
38, 20, 15, 1, 22, 265, 2, 19, 2, 1, 48, 3, 1, 14, 1, 17, 4, 1], [22, 1, 33, 1, 1, 52, 2, 67, 1, 2
6, 7, 1, 68, 44, 15, 1, 14, 1], [28, 23, 1, 21, 19, 123, 15, 1, 46, 1, 66, 11, 45, 1, 156, 129, 5,
1, 6, 80, 1, 9, 1], [26, 190, 11, 1, 13, 3, 1, 35, 6, 1, 48, 38, 65, 1], [37, 51, 94, 1, 3, 2, 1,
4, 2, 1, 85, 1, 69, 80, 41, 37, 1, 270, 154, 1], [20, 11, 45, 1, 21, 61, 81, 1, 5, 41, 76, 5, 1, 4
7, 4, 1, 84, 1, 1, 16, 1, 71, 1, 92, 2, 1, 58, 16, 1, 113, 1, 1], [39, 1, 35, 16, 1, 14, 55], [181
, 3, 1, 20, 8, 1, 96, 20, 7, 1, 17, 10, 11, 1, 6, 36, 4, 1, 4, 2, 1, 26, 7, 27, 1], [21, 61, 81, 1
, 30, 22, 1, 21, 19, 123, 15, 1, 165, 31, 1, 8, 29, 1, 30, 22, 1], [7, 31, 31, 1, 40, 1, 91, 13, 1
, 104, 82, 57, 25, 31, 1], [14, 62, 1, 177, 8, 1, 14, 1, 22, 18, 1, 105, 52, 1, 3, 2, 27, 1, 25, 5
0, 1], [14, 15, 10, 1, 25, 1, 19, 1, 201, 103, 76, 57, 2, 1, 21, 61, 81, 1, 8, 29, 1, 60], [6, 30,
11, 1, 21, 61, 81, 1, 35, 6, 1, 135, 24, 20, 74, 2, 1, 60, 1], [92, 2, 1, 12, 4, 1, 33, 10, 80, 10
, 1, 9], [155, 1, 133, 38, 1, 1, 12, 98, 1, 8, 29, 1, 3, 2, 62, 1, 14, 1, 294, 1], [7, 99, 1, 22,
1, 17, 4, 1, 4, 2, 1], [104, 82, 57, 25, 31, 1, 52, 2, 253, 1], [28, 23, 1, 21, 19, 123, 15, 1, 16
, 1, 58, 16, 1, 35, 16, 1, 3, 15, 1, 17, 13, 1], [12, 4, 1, 69, 80, 41, 37, 1, 3, 2, 45, 1, 80, 12
0, 13, 1], [74, 32, 1, 14, 15, 10, 1, 68, 19, 2, 1, 140, 3, 38, 1], [34, 2, 1, 34, 2, 1, 13, 3, 1,
42, 1, 131, 144, 1, 133, 50, 1, 99, 1, 2, 103, 1, 14, 1], [3, 7, 1, 141, 8, 1, 71, 8, 1], [58, 16,
1, 270, 15, 27, 1, 49, 1, 12, 4, 1, 185, 101, 231, 45, 1, 9, 1], [68, 32, 11, 53, 8, 1, 105, 18, 1
, 9, 1, 101, 43], [68, 44, 15, 1, 14, 1, 26, 7, 1, 1], [69, 80, 41, 37, 1, 3, 2, 27, 1, 40, 1, 91,
```

13, 1, 104, 82, 57, 25, 31, 1, 28, 23, 1, 42, 111, 1], [13, 3, 1, 141, 287, 1, 48, 3, 1, 3, 2, 1,
119, 1, 37, 51, 94, 1], [114, 96, 1, 114, 15, 1, 13, 3, 1, 35, 6, 1, 48, 3, 1, 3, 2, 45, 1], [39,
1, 19, 7, 1, 6, 30, 11, 1, 21, 61, 81, 1, 66, 4, 1, 85, 27, 1], [78, 95, 1, 77, 1, 133, 10, 1, 2,
103, 1, 14, 1, 4, 2, 1], [1, 89, 8, 15, 1, 46, 1, 141, 287, 1, 48, 3, 1, 3, 2, 1, 5, 3, 62, 1, 14,
1, 12, 4], [37, 51, 94, 1, 16, 1, 13, 3, 1, 19, 7, 1, 3, 2, 45, 1], [25, 32, 1, 1, 1, 1, 12, 4, 1,
138, 5, 7, 1, 9, 1], [92, 2, 1, 66, 11, 45, 1, 196, 11, 172, 2, 1, 3, 2, 25, 65, 107, 1, 101, 43,
1], [18, 36, 1, 8, 29, 1, 3, 6, 1, 58, 16, 1, 25, 18, 5, 1, 22, 107, 1], [26, 7, 1, 84, 32, 1, 19,
108, 1, 196, 130, 97, 15, 1, 14, 1], [20, 5, 1, 13, 18, 1, 63, 1, 92, 2, 1, 68, 32, 11, 2, 1, 3, 2
, 1, 5, 3, 62, 1, 14, 1], [3, 6, 1, 11, 3, 1, 157, 142, 7, 1, 14, 1, 99, 107, 1, 28, 23, 1, 21, 19
, 123, 15, 1, 40, 1, 58, 16, 1], [22, 37, 1, 11, 3, 1, 49, 1, 5, 6, 16, 1, 68, 19, 2, 1, 5, 41, 76
, 5, 1, 67, 1], [135, 24, 20, 74, 2, 1, 19, 1, 6, 41, 36, 74, 1, 21, 61, 81, 1, 8, 29, 1, 60, 1],
[112, 1, 22, 2, 1, 173, 86, 1], [174, 12, 112, 1, 269, 1, 7, 73, 1], [20, 104, 2, 1, 34, 69, 75, 1
, 30, 43, 1, 67, 1, 12, 4, 1, 132, 6, 32, 1], [173, 86, 1, 6, 1], [6, 153, 4, 1, 19, 7, 1, 66, 4,
1, 101, 43, 1], [20, 5, 1, 5, 41, 76, 5, 1, 16, 1, 19, 108, 1, 45, 24, 37, 1, 30, 1, 71], [57, 32,
1, 84, 1, 124, 24, 6, 1, 30, 22], [22, 37, 1, 7, 145, 1, 8, 29, 1, 22, 4, 1], [21, 23, 1, 6, 153,
4, 1], [13, 56, 18, 1, 19, 1, 18, 2, 41, 46, 5, 1, 6, 30, 11, 1, 124, 24, 6, 1, 9, 1], [6, 30, 11,
1, 3, 82, 267, 17, 1, 196, 15, 2, 5, 1, 9, 1, 26, 54, 33, 24, 1], [3, 1, 10, 128, 1, 9, 1, 92, 2,
1, 21, 6, 1, 11, 3, 1, 117, 97, 74, 2, 1, 8, 29, 1, 66, 4, 1, 109, 45, 1], [164, 44, 15, 1, 141, 1
14, 1, 96, 20, 7], [86, 1, 15, 1, 9, 1, 39, 1], [72, 79, 1, 78, 95, 1, 5, 7, 83, 1, 46, 1, 8, 29,
1, 22, 1, 2, 33, 1, 9, 1], [12, 4, 1, 68, 19, 2, 1, 69, 44, 5, 1, 9, 1, 20, 5, 1, 13, 18, 1, 19, 1
], [68, 19, 2, 1, 56, 38, 8, 1, 9, 1, 20, 8, 19, 1], [39, 1, 35, 6, 1, 3, 33, 1, 2, 32, 1, 17, 4,
1, 4, 2, 1], [12, 4, 1, 20, 11, 45, 1, 124, 41, 73, 1, 3, 2, 45, 1, 5, 42, 1, 14, 55, 1], [28, 24,
1, 96, 20, 7, 1, 6, 41, 34, 31, 1, 3, 2, 1, 47, 4, 1], [35, 6, 1, 3, 33, 1, 9, 1, 101, 43, 1], [35
, 6, 1, 135, 24, 20, 74, 2, 1, 40, 1, 21, 44, 15, 1, 13, 3, 1, 56, 50, 8, 1, 3, 111], [20, 104, 2,
1, 48, 125, 17, 1, 35, 16, 1, 1, 14, 36, 1, 9, 1], [26, 54, 32, 24, 1, 63, 59, 1, 63, 59, 1, 104,
82, 57, 25, 31, 1, 28, 24, 1, 21, 19, 123, 15, 1, 2, 235, 44, 15, 2, 1, 3, 2, 27, 1, 40, 1, 91, 13
, 1], [18, 2, 239, 1, 19, 1, 63, 43, 1, 70, 38, 20, 15, 1, 59, 3, 64, 5, 1, 47, 109, 65, 1, 21, 32
, 7, 118, 34, 31, 1, 16, 1, 204, 25, 91, 57, 2, 1, 19], [163, 12, 69, 4, 1, 69, 80, 41, 37, 1, 3,
2, 27, 1, 40, 1, 91, 13, 1, 119, 44, 11, 1], [21, 68, 1, 238, 1, 14, 1, 12, 4, 1, 294], [14, 15, 1
0, 1, 25, 50, 1, 27, 1, 72, 83, 16, 1, 279, 4, 118, 1, 58, 52, 1, 91, 4, 1], [85, 1, 84, 1, 6, 30,
11, 1, 5, 7, 83, 118, 2, 1, 9, 1], [28, 24, 1, 14, 15, 10, 1, 46, 1, 44, 130, 1, 21, 61, 81, 1, 8,
29, 1, 60, 1], [65, 13, 1, 14, 13, 1, 52, 2, 67, 1, 26, 7, 27, 1, 6, 30, 11, 1, 21, 61, 81, 1, 19,
7, 1, 66, 4], [26, 7, 1, 84, 1, 38, 37, 25, 6, 1, 14, 1], [71, 1, 45, 24, 37, 1, 30], [21, 68, 1,
39, 1, 13, 56, 18, 1, 66, 11, 45, 1, 114, 5, 36, 1, 9, 1], [12, 4, 1, 28, 24, 1, 13, 3, 1, 19, 7,
1, 3, 2, 119, 107, 1], [70, 38, 20, 15, 1, 110, 1, 50, 17, 44, 139, 5, 1, 20, 11, 45, 1, 19, 7, 1,
60, 1], [6, 153, 4, 1, 217, 10, 1, 89, 38, 1, 32, 28, 1], [71, 1, 17, 1, 22, 1, 210, 19, 1, 162, 1
], [26, 54, 33, 23, 1, 19, 7, 1, 19, 1, 63, 43, 1, 37, 25, 6, 1, 8, 29, 1, 52, 2, 67, 1, 106, 34,
75], [15, 64, 216, 1, 13, 18, 1, 9, 1, 39, 1], [184, 148, 7, 1, 148, 253, 1, 12, 4, 1, 9, 1, 28, 2
3, 1, 21, 19, 123, 15, 1, 165, 31, 1, 14, 27, 1, 49, 1], [14, 15, 10, 1, 44, 96, 114, 1, 117, 161,
56, 86, 1, 8, 29, 1, 60, 1], [35, 6, 1, 135, 24, 20, 74, 2, 1, 48, 24, 1, 60, 1], [26, 54, 33, 23,
1, 74, 37, 32, 1, 16, 1, 68, 214, 142, 148, 1, 14, 1, 17, 121, 1, 28, 53], [72, 79, 1, 26, 7, 1, 1
6, 1, 6, 30, 11, 1, 149, 19, 57, 11, 1, 9, 1], [66, 11, 45, 1, 58, 52, 1, 9, 1, 28, 23, 1, 21, 19,
123, 15, 1, 46, 1], [21, 61, 248, 1, 48, 125, 17, 1, 40, 1, 91, 13, 1, 104, 82, 57, 25, 31, 1], [1
2, 4, 1, 202, 144, 2, 1, 52, 1, 13, 18, 1, 9, 1], [3, 77, 1, 84, 1, 5, 7, 83, 1, 99, 1, 13, 3, 1,
34, 2, 1, 28, 1], [39, 1, 12, 4, 1, 6, 3, 25, 5, 1, 9, 1], [52, 2, 67, 1, 49, 1, 37, 57, 1, 14], [
141, 287, 1, 89, 10, 1, 17, 4, 1, 1, 121, 126, 1], [26, 7, 1, 6, 153, 4, 1, 14, 1, 4, 2, 1], [34,
2, 1, 34, 2, 1, 53, 44, 96, 41, 15, 1, 3, 2, 27, 1, 16, 1, 77, 1, 74, 42, 1, 47, 109, 1, 2, 33, 1]
, [6, 154, 1, 7, 188, 118, 2, 1, 14, 1, 26, 7, 1], [12, 4, 1, 87, 17, 10, 127, 1, 9, 1, 39, 1], [2
5, 32, 1, 101, 43, 1, 12, 4, 1, 6, 218, 4, 1, 217, 10, 1, 89, 97, 1, 9, 1], [71, 1, 140, 24, 75, 1
, 162, 1], [72, 79, 1, 6, 30, 11, 1, 173, 44, 52, 1, 22, 1, 2, 33, 1, 9, 1], [28, 24, 1, 13, 3, 1,
19, 7, 1, 3, 2, 1, 119, 1], [167, 17, 116, 2, 1, 46, 1, 63, 43, 1, 77, 1, 14, 15, 10, 1, 48, 3, 1,
3, 2, 1, 119], [21, 68, 1, 69, 80, 41, 37, 1, 3, 2, 1, 210, 18, 1, 80, 18, 1, 16, 1], [101, 43, 1,
85, 27, 1, 28, 53, 1, 6, 30, 11, 1, 103, 51, 18, 1, 49, 1, 9, 1], [3, 6, 1, 11, 3, 1, 69, 80, 41,
37, 1, 14, 1, 99, 13, 55, 1, 28, 24, 1], [47, 51, 97, 1, 46, 1, 34, 69, 75, 1, 14, 27, 1, 25, 97,
1, 9, 1, 12, 4, 1], [21, 23, 1, 39, 1, 115, 8, 51, 5, 1, 12, 98, 1, 8, 29, 1, 22, 1, 2, 33, 1, 194
, 148, 39, 1], [25, 3, 43, 1, 36, 53, 114, 1, 40, 1, 19, 201, 10, 1, 9, 1], [26, 7, 1, 48, 125, 17
, 1, 3, 2, 27, 1, 46, 1, 20, 11, 100, 1, 113, 2, 1, 12, 98, 1, 10, 55, 62, 1, 14, 1], [86, 1, 247,
52, 1, 19, 7, 1, 66, 4, 1, 26, 7, 27, 1], [26, 7, 1, 6, 30, 11, 1, 44, 90, 41, 15, 1, 14, 1], [64,
93, 8, 1, 59, 3, 15, 1, 21, 18, 27, 1, 22, 18, 1, 178, 161, 10, 1, 35, 16, 1, 14, 1, 17, 121, 1]]
```
[[ 26   7   1 ...   0   0   0]
 [174  12   5 ...   0   0   0]
 [121 126   1 ...   0   0   0]
 ...
 [ 86   1 247 ...   0   0   0]
 [ 26   7   1 ...   0   0   0]
 [ 64  93   8 ...   0   0   0]]
```

Generating Contextual vectors for each Characters using Gensim

In [29]:

```
model_char = Word2Vec(char_sentence,min_count=1,size=300,window=10,negative=7)

model_char[char_sentence[0][3]]
```

Out[29]:

```
array([ 1.21993147e-01, -1.10313252e-01, -1.30273506e-01, -3.94060090e-02,
        1.83083743e-01,  1.81165591e-01, -1.01802005e-02, -1.07633255e-01,
       -4.86547034e-03,  6.45240098e-02, -1.11426905e-01, -6.41342402e-02,
        2.73756795e-02,  6.91808835e-02, -1.22568630e-01, -1.81447521e-01,
       -1.02773778e-01, -2.39948615e-01,  8.86167586e-02,  1.29263267e-01,
       -9.35102347e-03,  5.79653531e-02,  1.18623421e-01, -1.65647119e-01,
        3.83384638e-02,  1.15360272e-04,  8.73810053e-02,  1.09993100e-01,
        5.28741963e-02, -2.26853006e-02, -1.08811110e-02, -1.01842873e-01,
        3.02893460e-01, -1.84737548e-01,  2.02915907e-01,  1.38720155e-01,
       -7.60268494e-02, -1.24955967e-01, -5.48529290e-02,  8.46853927e-02,
        8.45115110e-02,  6.39002845e-02,  8.40898752e-02,  4.45679128e-02,
        2.56559700e-02,  1.25743330e-01, -7.21911862e-02,  2.33478427e-01,
       -7.52621293e-02,  1.91711739e-01,  1.58869237e-01,  1.99997984e-02,
        6.79100901e-02,  7.66174868e-02, -1.00557469e-01,  7.89843798e-02,
        6.42207637e-02, -9.05053690e-02,  1.13520538e-02, -7.62807056e-02,
        9.83518809e-02,  9.76374093e-03, -5.28837815e-02,  2.16062114e-01,
        1.42926380e-01,  4.26006019e-02,  1.18184380e-01, -1.53870732e-01,
       -7.40618855e-02,  2.56193221e-01, -6.70639798e-02, -1.46585792e-01,
        1.98983505e-01, -1.30845413e-01,  2.39862591e-01,  8.38058069e-02,
       -2.90593170e-02, -2.08037104e-02,  1.62365913e-01,  3.02826008e-03,
        1.58503011e-01, -4.31574807e-02, -2.19712388e-02,  2.36069486e-01,
        5.19308485e-02, -9.57439840e-02, -1.70956269e-01, -8.10025185e-02,
       -5.34375161e-02, -2.01358616e-01,  9.66666862e-02, -3.91561212e-03,
       -1.62355214e-01, -1.09965406e-01,  1.58787936e-01, -9.30843726e-02,
       -4.61367518e-02, -1.27853736e-01,  2.49510005e-01,  1.26747414e-01,
       -9.68698710e-02, -6.36859983e-02,  4.21200274e-03,  1.68576598e-01,
       -2.43299827e-01, -9.77287143e-02,  5.84379919e-02,  1.03402577e-01,
        1.34539619e-01,  6.83930814e-02, -2.05086693e-02, -9.10980627e-03,
       -2.08977133e-01, -6.53326362e-02, -9.41073149e-02, -4.61540557e-03,
        9.21099931e-02, -1.08371057e-01, -1.29204154e-01, -1.08795874e-01,
        1.00092746e-01, -1.06096335e-01,  3.88183072e-02,  1.38638858e-02,
        4.67967615e-02, -1.44969076e-01, -1.94044411e-02,  2.05776155e-01,
       -6.47926554e-02,  1.53660685e-01,  9.31290761e-02, -3.66660506e-02,
        1.23362444e-01, -1.18649922e-01,  9.36278701e-02, -1.79246038e-01,
       -1.72476079e-02,  3.78856622e-02,  9.94334817e-02,  1.51120529e-01,
       -9.69417095e-02, -8.14318731e-02,  1.91145837e-01,  2.00515017e-01,
       -2.11726382e-01, -6.76326454e-02,  1.43753514e-01, -3.65509838e-02,
        1.00056402e-01, -1.44598996e-02, -1.54839426e-01, -9.07015335e-03,
        2.66098920e-02, -1.84043705e-01,  1.29206227e-02,  7.41102993e-02,
       -4.80348766e-02,  2.26078048e-01,  1.41563872e-02, -1.67682484e-01,
       -2.25766841e-02, -1.71165973e-01,  1.05858006e-01, -7.08995461e-02,
       -1.50186494e-02, -5.97628877e-02,  1.36803640e-02, -5.84134087e-02,
       -1.18909806e-01,  2.17391610e-01,  1.54333711e-01,  2.23648563e-01,
       -4.59227264e-02, -9.27598625e-02, -1.35940744e-03, -5.84404245e-02,
       -1.99867040e-02, -1.02920100e-01, -7.87906572e-02,  3.35394293e-02,
       -7.43422210e-02,  1.88426878e-02,  5.71751371e-02,  1.40602782e-01,
       -9.83882397e-02, -4.89132553e-02,  1.30481362e-01, -6.30898625e-02,
        1.18490467e-02,  4.37159054e-02,  7.23276511e-02,  7.93272853e-02,
       -1.81037337e-01, -1.65963024e-02,  2.47883096e-01, -9.12079066e-02,
       -1.04804942e-02, -2.43660510e-02,  1.31838620e-01, -6.30080849e-02,
        1.08443186e-01,  5.43368831e-02, -9.19820517e-02, -7.07667470e-02,
       -9.49905440e-02,  5.00560962e-02,  7.20967725e-02, -1.75914720e-01,
       -3.48836593e-02, -8.43426585e-02, -1.82541013e-01,  6.48018420e-02,
       -4.11266387e-02, -6.58227429e-02, -9.74878296e-02, -1.97788477e-02,
       -1.41054541e-01, -1.07633263e-01,  2.02305801e-03, -9.20148790e-02,
        4.54239398e-02,  4.75174412e-02,  9.04158130e-02, -1.01035595e-01,
        1.64741635e-01,  5.89199103e-02, -1.07898600e-01,  4.41601388e-02,
        3.43829170e-02,  1.19745240e-01, -1.48822926e-02,  4.37834077e-02,
       -1.93474799e-01, -1.45255938e-01, -1.19884983e-01,  3.10771950e-02,
        4.23767269e-02, -2.56214943e-02, -1.99344292e-01,  6.28848597e-02,
       -1.10504344e-01, -2.96901446e-02, -2.49252841e-02,  3.66446450e-02,
       -1.30162090e-01, -2.25651413e-01,  2.63994128e-01, -2.58556843e-01,
       -3.76910367e-03, -3.22397426e-02,  2.01516032e-01, -7.54462406e-02,
       -8.00251886e-02, -1.31505519e-01, -1.16511844e-01,  5.41053005e-02,
        1.12136722e-01,  9.23504606e-02, -1.39910802e-01,  9.98546332e-02,
        6.26356676e-02, -2.09994782e-02,  3.35504189e-02, -3.67348969e-01,
        1.27838263e-02,  2.24306628e-01, -1.21552532e-03,  2.69848466e-01,
       -5.89464344e-02,  1.24821424e-01,  1.12529211e-01,  1.44803310e-02,
        2.52961874e-01, -4.91416007e-02,  4.47416957e-03,  1.74839437e-01,
        1.06300928e-01, -1.25162870e-01, -1.77532464e-01,  9.00463834e-02,
        2.37220019e-01, -1.20891862e-01,  2.01087967e-01,  1.88556880e-01,
        2.87562519e-01, -7.61249438e-02, -2.86411494e-01, -9.79609713e-02,
       -4.70385253e-02,  3.21568325e-02,  2.65199970e-02,  4.02035341e-02,
       -5.56630827e-02,  2.72070598e-02,  5.58575131e-02,  5.24983779e-02,
```

```
       -7.95045421e-02, -2.66713388e-02,  1.78423330e-01,  4.01140265e-02],
      dtype=float32)
```

In [30]:

```
word_vectors_char = model_char.wv
len(word_vectors_char.vocab)
```

Out[30]:

```
317
```

creating Embedding matrix for all characters to store vectors of each character

In [0]:

```
embedding_matrix_char = np.zeros((len(word_vectors_char.vocab), 300))
for word, i in t_char.word_index.items():
  if word in word_vectors_char.vocab and word in t_char.word_index.keys():
    embedding_vector_char = model_char[word]
    if embedding_vector_char is not None:
      embedding_matrix_char[i] = embedding_vector_char
```

Model for Characters

Model for Non-Contextual-vectors without a Embedding Matrix for Characters

Intialize Vectors for Each word with random vectors and Learn those Vectors for characters

In [32]:

```
from keras.layers import Dense, Dropout, Embedding, LSTM, Bidirectional
sequence_input_char = Input(shape=(45,))
# Embedding Layer
embedded_sequences_char = Embedding(len(word_vectors_char.vocab),output_dim=150,input_length=45)(s
equence_input_char)
# LSTM  Layer using Sequence from Backward-last
# LSTM  Layer using Sequence from Forward Sequence
lstm_2_model_char = Bidirectional(LSTM(64,return_sequences=True,dropout=0.5))
(embedded_sequences_char)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:541: The name tf.placeholder is deprecated. Please us
e tf.compat.v1.placeholder instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:4432: The name tf.random_uniform is deprecated. Pleas
e use tf.random.uniform instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:3733: calling dropout (from
tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future
version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
```

In [33]:

```
lstm_2_model_char.shape
```

Out[33]:

```
TensorShape([Dimension(None), Dimension(None), Dimension(128)])
```

In [0]:

```
flatten_layer_1 = Flatten()
X_2_char = flatten_layer_1(lstm_2_model_char)
```

In [35]:

```
X_2_char.shape
```

Out[35]:

```
TensorShape([Dimension(None), Dimension(None)])
```

Training Bi-directional LSTM using a Contextual vectors obtained from gensim for characters

In [36]:

```
embedded_sequences_char =
Embedding(len(word_vectors_char.vocab),output_dim=300,weights=[embedding_matrix_char],input_length
=45,trainable=False)(sequence_input_char)
lstm_out_model_backward_1_char = Bidirectional(LSTM(64,return_sequences=True,dropout=0.5))
(embedded_sequences_char)
flatten_layer_2 = Flatten()
X_char = flatten_layer_2(lstm_out_model_backward_1_char)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:190: The name tf.get_default_session is deprecated. P
lease use tf.compat.v1.get_default_session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:197: The name tf.ConfigProto is deprecated. Please us
e tf.compat.v1.ConfigProto instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:203: The name tf.Session is deprecated. Please use tf
.compat.v1.Session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:207: The name tf.global_variables is deprecated. Plea
se use tf.compat.v1.global_variables instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:216: The name tf.is_variable_initialized is
deprecated. Please use tf.compat.v1.is_variable_initialized instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:223: The name tf.variables_initializer is deprecated.
Please use tf.compat.v1.variables_initializer instead.
```

In [37]:

```
X_char
```

Out[37]:

```
<tf.Tensor 'flatten_2/Reshape:0' shape=(?, ?) dtype=float32>
```

In [38]:

```
X_2_char
```

Out[38]:

```
<tf.Tensor 'flatten_1/Reshape:0' shape=(?, ?) dtype=float32>
```

text Processing using keras Tokenizer for Words

In [39]:

```
maxLen = len(max(list_sentences, key=len).split())
maxLen
```

Out[39]:

11

train the word tokenizer using first 400 words using words

In [40]:

```
from keras.preprocessing.text import Tokenizer

t = Tokenizer()
t.fit_on_texts(list_sentences[:400])
vocab_size = len(t.word_index) + 1
# integer encode the documents
encoded_docs_train = t.texts_to_sequences(list_sentences[:400])
print(encoded_docs_train)
# pad documents to a max length of 4 words
max_length = 13
padded_docs_train = sequence.pad_sequences(encoded_docs_train, maxlen=max_length, padding='post')
print(padded_docs_train)
```

```
[[20, 60, 61, 28, 291, 31, 3, 132], [133, 61], [7, 11, 31, 1], [77, 182, 69, 292, 106, 89, 90], [4
4, 18, 18, 293, 45, 21, 1], [91, 183, 107, 294, 108, 32, 9], [41, 295, 1, 2], [184, 2, 109, 46, 3,
47, 21, 1], [92], [11, 60, 134, 296, 185, 1], [11, 37, 186, 26, 16], [62, 37, 297, 6, 51], [298, 1
87, 55, 52, 1, 56, 78, 10], [24, 41, 188, 1, 2], [110, 78, 22, 57, 133], [189, 5, 190, 12, 79, 191
, 4, 111], [11, 9, 299, 300, 38, 18, 3, 16, 5], [13, 63, 64, 192, 3, 193, 62], [29, 39, 41, 17, 4,
19, 93], [94, 194, 11, 1], [14, 301, 12, 58, 302, 33], [2, 135, 195, 303], [2, 42, 1], [13, 112, 2
7, 113, 1, 43, 70], [91, 35, 22, 304, 9, 38, 90], [5, 40, 305, 26, 3, 136, 21], [20, 60, 61, 28, 3
06, 3, 65], [114, 34, 79, 191, 6, 196, 95], [13, 26, 42, 197, 18, 198], [96, 52, 115, 48, 307, 3,
116], [41, 17, 3, 117], [71, 308, 20, 32, 309], [49, 7, 80, 3, 33], [2, 199, 310, 1], [20, 60, 118
, 200, 97, 10, 11, 311], [312, 12, 98, 137, 313, 56], [25, 39, 15, 41, 17, 81, 4], [46, 3, 47, 21,
5, 2, 119, 23], [314, 13, 27, 3, 6, 315, 72, 4, 25], [36, 24, 201, 316, 202, 4, 99], [203, 317, 18
, 204, 10, 1, 318, 40, 319, 3, 320, 21], [40, 27, 15, 3, 1, 7, 35], [53, 37, 58, 31, 205, 107, 72]
, [8, 26, 321, 322, 16, 323], [23, 184, 34, 324], [36, 73, 4, 138, 2], [7, 206, 1], [20, 24, 15, 7
3, 207, 4, 21, 1, 18, 18], [5, 36, 325, 45, 3, 46, 72], [13, 14, 17, 326], [5, 24, 48, 327, 328, 6
, 196, 1], [92, 329, 16, 139], [330, 100, 23, 140, 120, 3, 101], [2, 96, 208, 4, 331], [14, 121, 4
8, 141, 332, 12, 42, 3, 10], [209, 210], [82, 333, 11, 16], [142, 10, 89, 66, 138, 143], [2, 42, 1
, 5, 122, 11, 66, 19], [54, 92], [143, 30, 144, 24, 66], [7, 211, 40, 334, 3, 1], [38, 335], [212,
212, 145, 213, 336, 23], [25, 77, 214, 4], [14, 83, 8, 337, 16], [338, 339, 215], [23, 84, 101], [
2, 53, 340, 146], [54, 144, 192, 216, 116, 5], [102, 110, 59], [123, 341, 106, 217, 147, 59], [203
, 218, 14, 17, 10, 33, 139, 219, 45, 3, 21], [85, 27, 22, 3, 4, 25], [35, 12, 40, 124, 3, 220], [4
9, 31, 342, 23, 343, 26, 6, 221], [344, 103, 28, 1, 222], [223, 10, 224, 12, 148, 10, 85, 39, 41,
1], [40, 63, 3, 345, 137], [53, 22, 225, 58, 346, 27, 4, 347, 1], [20, 60, 13, 110, 149, 16], [29,
125, 34, 348, 4, 126, 1, 150], [39, 10, 151, 58, 127, 33], [3, 349, 32, 67, 115, 350], [152, 84, 6
, 51, 70], [5, 18, 18, 351, 26, 45, 17, 226, 1], [128, 153, 14, 66, 99], [227, 35], [114, 26, 28,
228, 72, 37, 2, 42, 1], [24, 9, 352, 22, 57, 63, 40, 17, 6, 68], [50, 54, 154, 71, 4, 63, 19], [13
, 112, 205, 220, 1], [353, 229, 229, 74, 29, 354, 112, 355, 78, 22, 57], [230, 145, 111], [5, 23,
8, 100, 4, 19, 75], [7, 14, 231, 230, 145, 19], [14, 22, 83, 48, 30, 356, 9, 26, 357, 358], [150,
359, 232, 1, 91, 35, 48], [86, 360, 16, 11, 9], [2, 95, 233, 20, 76, 10, 361, 234], [235, 3, 155,
21, 67, 56], [54, 5, 64, 362, 107, 21], [363, 15, 77, 86, 44, 1, 61], [39, 48, 364, 365, 6, 366, 1
], [50, 5, 7, 14, 231, 156, 1, 367, 157], [2, 86, 236, 1], [30, 7, 44, 237, 31, 3, 132], [123, 2,
30, 87, 368, 46, 21, 1], [27, 4, 19], [55, 52, 13, 26, 42, 198], [14, 238, 136, 3, 369], [36, 24,
15, 2, 239, 1], [370, 35], [8, 26, 135, 371, 1, 61, 43], [39, 73, 34, 240, 46, 9, 3, 47, 21, 1], [
372, 30, 158, 373, 1, 2], [30, 37, 7, 44, 8, 31, 156, 1], [50, 54, 2, 42, 1], [36, 374, 154, 375,
34, 1], [14, 121, 48, 141, 376, 10], [24, 9, 8, 159, 65], [223, 15, 39, 188, 1], [5, 2, 103, 28, 1
], [377, 38, 241, 24, 17, 4, 126, 16], [55, 52, 242, 108, 32, 9], [8, 31, 43], [32, 243, 22, 57, 7
4, 20, 5], [47, 244, 2, 42, 1], [378, 86], [36, 39, 69, 12, 107, 245, 1, 379], [380, 160, 7, 80, 3
, 33], [7, 44, 120, 63, 381, 161, 382, 1], [383, 6, 51, 62], [5, 20, 32, 246, 55, 384], [36, 14, 4
5, 3, 47, 245, 1], [385, 5, 386, 4, 99], [247, 248], [5, 7, 37, 387, 388, 27, 162, 1], [249, 60, 1
3, 18, 9, 46, 9, 3, 389, 2], [390, 102, 250, 391, 163], [8, 100, 162, 1, 7, 35], [103, 28], [392,
9, 393, 10, 251, 58, 394, 4, 395, 1], [53, 15, 88, 164, 1], [74, 43], [129, 47, 19, 43], [39, 41,
17, 6, 126, 1, 87], [25, 8, 396, 4], [252, 397, 398, 12, 204, 165, 146], [50, 399, 43], [7, 2, 1],
[2, 25, 36, 124, 6, 81, 4], [35, 253, 254, 3, 1], [49, 7, 80, 3, 33], [30, 49, 7, 80, 3, 33], [123
, 255], [256, 400, 6, 51], [209, 210, 2, 27, 113, 1, 62], [153, 11, 16, 5], [93], [401, 40, 46, 9,
3, 257, 2, 166], [14, 83, 158, 16], [402], [248, 28], [108, 27, 32, 9, 94, 79], [13, 18, 9, 46, 3,
65, 258, 56, 78, 37], [167, 259, 29, 39, 41, 17, 4, 19], [84, 101, 67], [228, 114, 34, 4, 260, 1,
403, 35], [11, 168, 12, 404, 9, 1], [83, 48, 169, 6, 51], [50, 7, 170, 34, 405, 4, 19, 171], [53,
15, 406, 2, 1], [50, 5, 407, 408, 60, 409, 261, 261, 100, 4, 19, 75], [86, 14, 33, 5], [25, 13, 25
4, 410, 4], [23, 49, 411, 262], [82, 64, 236, 1, 5], [20, 76, 9, 82, 125, 1], [14, 55, 52, 157], [
263, 7, 140, 4, 260], [412, 413], [7, 44, 8, 414, 172, 19, 1, 5], [8, 129, 65], [25, 415, 206, 4],
```

```
[49, 31, 171, 12, 27, 155, 221], [7, 44, 416, 104, 417], [418, 28, 10, 80, 33, 160], [250, 419, 42
0], [264, 11, 16], [421, 422, 2, 1, 125, 423, 265, 9], [11, 15, 266, 267, 16], [424], [67, 237, 42
5, 105, 252, 28, 12], [2, 426, 427, 1, 130, 44, 15], [95, 428, 28, 118, 97, 429], [32, 243, 22, 57
, 133, 430], [2, 67, 79, 431, 6, 126, 105], [152, 268, 173, 70], [186, 432, 44, 156, 1], [23, 37,
8, 84, 105, 7, 44, 433, 104], [434, 435], [269, 174, 118, 97, 436], [8, 437, 1, 5, 270], [131,
147, 175, 6, 38, 71, 56, 6], [50, 54], [7, 164, 103, 271, 1], [438, 83, 16], [176, 37, 8, 31, 211,
1], [2, 30, 439, 234, 233], [96, 22, 96, 17, 59], [440, 183, 10, 24, 10, 41, 155, 68, 441, 12, 224
, 10], [5, 7, 8, 64, 272, 1], [177, 2, 119], [263, 7, 4, 442, 5], [115, 29, 178, 55, 52, 157], [87
, 158, 1, 2], [77, 241, 24, 15, 41, 17, 6, 68, 179, 12, 148, 15], [89, 2, 69, 443, 444], [13, 445,
17, 59], [100, 4, 19, 446, 5], [74, 67, 84, 101], [38, 2, 2, 162, 1, 7, 35], [447, 8, 448, 75], [1
1, 12, 449, 31, 3, 16], [450, 28, 46, 22, 273, 1], [30, 24, 15, 41, 66, 19, 54], [25, 38, 451, 6,
81, 4], [29, 131, 175, 147, 6, 38, 208, 56, 6], [452, 453, 17, 6, 81, 4, 2], [53, 15, 88, 8, 122,
1, 5], [454, 455, 59, 61], [25, 456, 27, 22, 3, 4, 166], [2, 90, 5, 256, 26, 175, 1], [273, 2, 457
, 1], [120, 37, 27, 22, 458, 25], [154, 459, 33, 139, 136], [167, 259], [460], [461, 247, 462], [1
29, 3, 65, 91, 18], [50, 122, 13, 124, 6, 68, 102], [29, 24, 463, 17, 4, 19, 128], [8, 26, 103, 44
, 1, 61], [23, 58, 464, 19, 53], [2, 85, 274, 1, 130, 97], [5, 2, 135, 6, 72, 4, 25], [14, 465, 9,
38, 82, 69, 1], [62, 275, 466, 6, 51], [5, 467, 468, 469, 1], [2, 267, 1, 5], [94, 470, 45, 471],
[40, 32, 1, 472, 3, 20, 76, 9], [2, 64, 35, 1, 7], [276, 87, 71, 56, 473], [474], [31, 27, 113, 43
], [8, 475], [111, 476, 4, 25, 477], [277, 129, 47, 19], [222, 28, 12, 278, 84, 1], [180, 172, 21,
1, 478, 12, 42, 6, 21, 75], [108, 32, 9, 197, 225, 242], [30, 49, 42, 3, 165, 1], [47, 244, 102, 2
5, 92, 4], [2, 29, 24, 479, 69, 1], [163, 50, 480, 481], [53, 15, 88, 90], [8, 482, 105, 23, 130,
279], [115, 280, 4, 25], [189, 5, 36, 24, 202, 4, 99], [29, 174, 55, 52, 181], [483, 484, 20, 32,
485], [276, 35], [13, 27, 6, 68], [93, 142, 10, 89, 66, 138], [486, 487, 53, 37, 70, 152, 268, 173
, 43], [77, 144, 24, 34, 281, 488], [8, 26, 489], [7, 14, 490, 12, 69, 282, 17, 3, 117], [166], [8
8, 2, 491, 1], [143, 23, 71, 492, 47, 19], [77, 182, 106, 493, 174, 168, 494, 1], [36, 24, 495, 45
, 117, 1], [179, 12, 73, 45, 3, 4, 21, 1], [11, 10, 238, 127, 283, 33], [280], [496, 270, 12, 177,
28, 3, 497], [8, 26, 214, 28, 1, 11, 10], [498, 499, 16], [30, 500, 501, 3, 4, 21, 16], [134, 18,
13, 26, 272, 502, 146], [8, 69, 172, 21, 1, 17, 503, 253], [168, 284, 271, 504, 22, 285, 12, 64, 9
8], [23, 505, 3, 75], [58, 110, 6, 116, 70], [93, 5, 2, 506, 193, 1, 62], [40, 159, 283, 65],
[227], [20, 39, 15, 73, 207, 4, 21, 1, 18, 18], [118, 95, 60, 46, 2, 232, 1, 70, 507], [180, 34, 1
81], [508, 150, 4, 25, 137], [2, 509, 27, 113, 1, 43], [28, 510, 59], [511, 512, 513, 514], [82, 5
15, 11, 16], [87, 516, 269, 195, 517, 161, 518], [124, 78, 22, 57, 74], [176, 11, 58, 31, 16], [21
5, 519], [520, 151, 521, 3, 33], [163, 522, 523, 524], [20, 76, 15, 125, 525, 526], [58, 527, 1],
[131, 286, 6, 51], [64, 1], [528, 88, 264, 1], [50, 54, 2, 28, 1], [153, 287, 117, 53, 54], [199,
529, 9, 287, 265, 106, 1, 2], [43, 30, 13, 42, 90], [32, 96, 52, 246, 3, 530, 5], [531, 165, 149],
[164, 1, 87], [23, 532, 34, 240], [61, 85, 533, 3, 102], [29, 13, 534, 1], [20, 76, 9, 82, 535, 53
6, 1], [29, 131, 286, 4, 19, 1, 537, 538, 539], [540, 13, 18, 67, 46, 3, 257, 2], [11, 37, 92, 1],
[218, 170, 23, 3, 541, 542, 32, 45, 543], [278, 64, 544, 235, 72, 545], [31, 546, 3, 16], [2, 547,
1, 5], [7, 79, 15, 251, 94, 548, 45, 6, 549, 1, 18, 18], [550, 79, 10, 551, 552, 9, 8, 127, 1], [5
53, 15, 11, 16], [128, 13, 112, 274, 66, 19], [40, 180, 288, 68, 5, 10, 554, 555], [35, 9, 38, 556
, 557], [2, 62, 266, 558, 173], [277], [29, 14, 169, 6, 68], [2, 86, 10, 217, 6, 72, 4], [123, 13,
559, 17, 6, 68, 43, 22, 57], [25, 120, 40, 560, 26, 3, 4], [14, 83, 48, 30, 561, 116], [13, 289, 2
84, 11, 17, 59], [93, 36, 89, 71, 47], [562, 20, 57, 142, 9, 13, 563, 564], [7, 11, 15, 565, 94, 5
66, 45, 219, 21, 1], [128, 2, 288, 185, 1], [23, 49, 8, 567, 75], [73, 34, 98], [568], [8, 26, 289
], [62, 30, 8, 569, 6, 51], [7, 170, 34, 570, 119], [571, 176, 104, 572], [36, 151, 573, 574, 33,
74], [575], [49, 85, 27, 213, 226, 255], [5, 55, 98, 56, 20, 76, 15], [11, 36, 290, 22, 285, 12, 3
, 16], [167, 13, 73, 6, 161, 14, 10], [36, 39, 15, 239, 281, 59], [29, 76, 55, 52, 181], [576, 63,
13, 18, 577, 109, 149], [179, 85, 578, 579, 11, 1, 200, 97, 23], [7, 37, 31, 1], [14, 121, 48, 190
, 12, 169, 6, 51], [50, 7, 34, 98, 5], [130, 27, 10, 580, 581, 105], [582, 78, 282, 63, 3, 4, 21],
[71, 13, 39, 15, 41, 17, 59], [2, 583, 6, 51, 70], [14, 121, 48, 141, 584, 585, 586], [160, 42, 10
4, 31, 132], [134, 18, 258, 10, 29, 587, 588, 56, 109, 1], [8, 122, 101, 589, 32, 187, 12, 140, 47
, 111], [23, 590, 262], [95, 114, 34, 29, 178, 591, 81, 4], [91, 88, 22, 279, 23, 592, 75], [49, 3
, 4, 593], [275, 594, 4, 19, 29], [30, 249, 12, 38, 80, 33, 5], [595, 12, 148, 10, 201, 24, 596, 1
7, 109], [30, 49, 38, 290, 33], [127, 14, 66, 99, 54], [40, 63, 104, 20, 32, 597, 598], [38, 2, 2,
6, 81, 4, 25, 599, 57], [171, 22, 600, 216, 7, 601, 34, 602, 119, 67], [603, 11, 9, 159, 3, 65], [
2, 5, 177, 52, 178, 3, 65], [194, 35, 1], [74, 20, 43], [604, 1, 2]]
[[ 20  60  61 ...   0   0   0]
 [133  61   0 ...   0   0   0]
 [  7  11  31 ...   0   0   0]
 ...
 [194  35   1 ...   0   0   0]
 [ 74  20  43 ...   0   0   0]
 [604   1   2 ...   0   0   0]]
```

test the word tokenizer using text sentences-words

In [41]:

```
from keras.preprocessing.text import Tokenizer

# integer encode the documents
encoded_docs_test = t.texts_to_sequences(list_sentences[400:])
print(encoded_docs_test)
# pad documents to a max length of 4 words
max_length = 13
```

```
padded_docs_test = sequence.pad_sequences(encoded_docs_test, maxlen=max_length, padding='post')
print(padded_docs_test)
```

```
[[25, 122, 4, 5], [412], [143, 29, 24, 463, 17, 4, 19], [47, 244, 102, 25, 524, 4], [20, 76, 9, 82
, 125, 536, 1], [208, 13, 14], [71, 6, 5, 87, 56], [94, 31, 28, 51, 37, 12, 23, 38, 32, 161], [7,
34, 98], [577, 550, 69, 5, 62], [31, 101, 76, 191, 3, 101], [110, 22, 57, 74], [168, 4, 95, 180, 7
8, 121], [11, 15, 31, 3, 16], [8, 31, 14, 83, 16], [38, 2, 1], [218, 258, 45, 3, 4], [554, 47, 19,
5], [74, 413], [20, 76, 12, 32, 34, 111], [2, 56, 109, 149], [139, 11, 86], [18, 18, 13, 26, 272,
235, 72, 4], [182, 479], [32, 10, 2, 1], [402, 211, 1, 43], [524, 4, 25], [56, 78, 22, 57, 74, 20]
, [13, 17, 6, 68, 71], [13, 14, 17, 109], [7, 27, 8, 31, 113, 70], [40, 63, 72, 4, 5], [9, 17, 6,
81, 4, 2], [71, 12, 13, 27, 109], [54, 2, 88, 1], [38, 82, 43], [281, 3, 55, 32, 246], [25, 342, 5
8, 4], [91, 35, 60, 38, 6, 81, 4], [55, 52, 4, 20, 76, 22, 32], [53, 52, 10, 77, 86, 28, 33],
[83, 15, 31, 3, 16], [163, 522, 248], [461, 480, 462], [179, 287, 117, 33, 2], [248], [122, 27, 11
3, 43], [91, 28, 12, 58, 75, 23], [176, 37, 286, 101], [53, 129, 3, 65], [50, 122], [44, 15, 8, 28
6, 1], [8, 1, 61], [1, 38, 177, 52, 3, 113, 266], [400, 69], [1, 7], [30, 40, 46, 9, 3, 47, 21, 1]
, [2, 86, 1, 91, 35, 15], [86, 271, 1, 450], [7, 14, 230, 19, 5], [2, 94, 109, 64, 98], [29, 69, 6
, 51], [14, 230, 1, 43], [14, 83, 22, 13, 59], [179, 73, 34, 226, 1], [353, 229, 229, 74, 29, 76,
355, 78, 22, 57], [367, 15, 85, 24, 596, 311, 12, 15], [133, 56, 78, 22, 57], [123, 4, 2], [11, 12
1, 48, 141, 548, 178, 173], [87, 37, 8, 1], [29, 11, 9, 31, 3, 16], [167, 259, 102, 62, 8, 31, 27,
113], [25, 37, 92, 4], [23, 75], [123, 7, 44, 82, 1], [2, 29, 13, 27], [24, 67, 94, 27, 16], [122,
288, 220], [23, 47, 262], [130, 27, 15, 85, 533, 3, 102, 460], [35, 1, 7], [298, 2, 1, 20, 76, 191
, 265, 10], [11, 499, 3, 16], [14, 83, 16], [130, 12, 4, 99, 36], [30, 25, 12, 8, 1], [82, 178, 1,
20, 76, 9], [396, 73, 22, 57, 74], [2, 284, 35, 1], [120, 37, 46, 235, 13, 18, 253], [7, 2, 135, 1
], [102, 10, 4], [66, 19, 143], [25, 122, 4, 5], [18, 18, 582, 78, 12, 63, 219, 21], [415, 206, 4,
25], [2, 1, 7], [54, 43, 2, 289, 288, 185, 1], [23, 569, 262], [30, 8, 47, 21, 1], [29, 13, 27, 6,
68], [142, 9, 85, 63, 11, 17, 6, 68], [123, 56, 6, 483, 484, 12], [43, 70, 36, 8, 124, 10, 1], [55
, 52, 56, 4, 331, 29], [224, 9, 287, 265, 106, 1, 2], [50, 7, 79, 45, 3, 47, 21], [22, 1], [25, 73
, 78, 9, 154, 45, 261, 4], [388, 27, 113, 62], [25, 8, 4], [39, 41, 118, 95, 169, 34, 4, 99]]
[[ 25 122   4 ...   0   0   0]
 [412   0   0 ...   0   0   0]
 [143  29  24 ...   0   0   0]
 ...
 [388  27 113 ...   0   0   0]
 [ 25   8   4 ...   0   0   0]
 [ 39  41 118 ...   0   0   0]]
```

Generating Contextual vectors for each word using Gensim

In [42]:

```
text_corpus_list = []
for i in tqdm(range(0,len(list_sentences))):
  text_corpus_list.append(list_sentences[i].split(' '))

model = Word2Vec(text_corpus_list,min_count=1,size=300,window=10,negative=7)

model[text_corpus_list[0][3]]
```

```
100%|██████████| 513/513 [00:00<00:00, 57268.12it/s]
```

Out[42]:

```
array([-2.42220610e-03,  4.93610452e-04, -6.61120226e-04,  7.16995634e-03,
        5.64625021e-03, -1.40183454e-03, -6.07511308e-03,  9.34665513e-05,
       -2.15099612e-03, -1.39897468e-03,  5.07910224e-03, -8.21334682e-03,
       -8.61909054e-03, -1.57792717e-02, -2.35181558e-03, -7.43804453e-03,
        4.61891759e-03,  9.29983612e-03, -4.68366640e-03,  9.98253282e-03,
       -8.08361080e-03,  1.63578950e-02,  6.59770239e-03, -3.16851493e-03,
       -5.35719795e-03, -2.65453709e-03,  7.43357046e-03,  9.44372173e-03,
        2.56812293e-03, -1.71816978e-03, -4.78364388e-03, -1.44159142e-03,
        1.47028826e-02, -2.33850512e-03, -5.27737522e-03, -9.24611930e-03,
       -5.81307523e-03, -2.59645935e-03, -3.81946913e-03, -7.43339444e-03,
       -1.09935682e-02, -6.02348126e-04,  8.41051992e-03,  1.02265915e-02,
        8.48548952e-03,  1.41392639e-02, -1.12624178e-02, -3.28660215e-04,
       -1.81452706e-02, -3.13560548e-03,  3.52020282e-03, -1.07596023e-03,
        5.07612154e-03, -1.10811293e-02, -1.73148215e-02, -5.07273618e-03,
        2.39941268e-03,  2.57887738e-03, -6.97041862e-03,  6.37871446e-03,
       -9.01096035e-04,  1.98995089e-03,  2.50159390e-03, -8.48824903e-03,
       -1.36565533e-03,  4.71079117e-03, -3.67265265e-03, -7.63907190e-03,
       -1.47886807e-03,  6.02755346e-04,  6.67242392e-04, -9.99522023e-03,
       -9.15703084e-03, -6.06095325e-03,  1.64924574e-03, -4.72751772e-03,
       -2.55099335e-03,  7.77347432e-03, -3.06482310e-03,  5.48781408e-03,
       -2.76335049e-03, -6.02990482e-03,  4.03290149e-03, -3.15191667e-03,
```

```
        2.63875292e-04,  2.29658862e-03, -6.80295099e-03, -3.25466320e-03,
        6.91934675e-03, -1.31550822e-02,  9.45823733e-03, -2.62784655e-03,
        9.72369511e-04, -8.90928134e-03,  4.11803136e-03,  5.77503536e-03,
       -2.54276162e-03,  1.73726827e-02,  1.83979571e-02,  2.74197804e-03,
        1.11050496e-03,  8.02706648e-03,  7.96348508e-03,  5.64116798e-03,
       -7.42508052e-03,  2.83476734e-03, -1.52637474e-02, -1.90845551e-03,
        3.78420064e-03, -2.19733827e-03,  4.06587357e-03,  1.18208751e-02,
        2.24330649e-03, -2.44761398e-03,  1.33829890e-02, -3.37974960e-03,
        3.46353627e-03,  1.22103300e-02, -2.25474732e-03,  1.72059797e-02,
       -6.10153237e-03, -2.04698648e-03, -1.80420396e-03,  2.60751974e-03,
       -6.50722068e-04, -8.40754714e-03,  8.42562225e-03, -4.12547868e-03,
        3.03893094e-03, -1.29441554e-02,  1.02980072e-02,  1.00927120e-02,
       -1.04191946e-02, -1.36369774e-02, -1.02132317e-02, -1.17663818e-03,
       -2.12589689e-02,  3.80436471e-03,  8.03209003e-03,  1.28482445e-03,
        9.94976796e-03,  7.39243627e-03, -1.25373458e-03, -2.17789272e-03,
        3.25844204e-03,  7.29388604e-03, -3.21698561e-03,  6.57964777e-03,
       -6.72012859e-04, -9.27483954e-04, -3.97627661e-03, -5.10359788e-03,
        1.54740289e-02, -6.56125788e-03, -1.06670624e-02,  3.59723810e-03,
       -7.95431528e-03,  2.94488831e-03, -1.51604170e-03,  1.57723180e-03,
       -9.01520904e-03,  3.27310269e-03, -2.02837912e-03, -8.43536295e-03,
        5.34541765e-03, -2.57086963e-03, -8.50801473e-04, -5.81249967e-03,
        5.13303978e-03,  8.91500153e-03,  1.13424554e-03, -2.32155784e-03,
        1.34139126e-02,  8.76493927e-04,  1.42433271e-02, -1.55500220e-02,
        2.33698776e-03,  1.36311157e-02, -1.05894506e-02,  3.51337669e-03,
        1.38913956e-03,  1.67255208e-03, -1.10942521e-03,  1.28550979e-03,
        2.05064216e-03,  1.49962050e-03,  7.71037943e-04, -1.04685929e-02,
        2.70938501e-03,  5.80777589e-04, -5.29541820e-03,  6.35246048e-03,
       -1.18880009e-03,  5.16680581e-03, -1.07462164e-02,  1.03773410e-02,
       -9.76963155e-03, -9.16029036e-04,  1.65033841e-03, -3.47411749e-03,
       -1.38163352e-02,  3.27256811e-03,  8.26219842e-03,  7.45753990e-03,
        1.65231188e-03,  7.58373039e-03,  1.12474263e-02, -3.97183327e-03,
        1.61189097e-03,  8.70520424e-04, -1.35723837e-02, -1.21422652e-02,
       -9.39480867e-03,  4.65860777e-03, -2.69457372e-03,  3.15765711e-03,
        1.97729282e-03,  7.19047035e-04,  9.05648060e-03,  3.94241419e-03,
        8.87261063e-04, -9.49591585e-03,  1.67015828e-02, -1.15929646e-02,
        7.60504277e-03, -2.97256256e-03, -8.66661780e-03,  1.60204638e-02,
        4.63305879e-03, -1.38984174e-02,  4.47696401e-03, -3.94259067e-03,
       -1.14787440e-03,  7.08216103e-03,  4.13231179e-03, -1.03747873e-02,
        1.32845147e-02,  5.76529047e-03,  1.02015585e-02,  8.26229155e-03,
        5.92443813e-03, -3.64892581e-03,  9.21485294e-03, -7.77794514e-03,
       -5.06661413e-03, -4.34656022e-03,  5.63607272e-03, -1.24262040e-03,
        6.19527139e-03,  7.57171714e-04,  1.21036032e-02,  5.48605574e-03,
        4.44334978e-03,  5.82705671e-03, -3.66460555e-03,  8.19592923e-03,
        3.26245464e-03,  5.00285299e-03, -5.42055676e-03, -1.84902095e-03,
       -2.08276650e-03, -5.71136922e-03, -8.86132661e-03, -5.28054405e-03,
        1.27600161e-02,  4.42114891e-03, -2.91214813e-03,  7.42883142e-03,
       -3.43291927e-03, -8.52774829e-03,  4.46944032e-03, -1.16875265e-02,
        5.01724146e-03, -6.89625181e-03,  2.08449620e-03, -4.76492220e-04,
       -1.37675583e-04,  3.36908961e-05,  6.50014030e-03,  4.64913575e-03,
       -3.65572232e-05, -2.34950887e-04,  6.61568064e-03,  2.72611785e-03,
        1.18737658e-02,  9.12035070e-03,  3.45922145e-03,  3.73071525e-03,
        1.04737664e-02,  3.65334190e-03, -1.83802936e-02,  1.12063847e-02,
        8.73979460e-03, -9.55113210e-03, -1.31206401e-02,  2.10051285e-03,
       -2.21665693e-03, -1.28623471e-03, -3.43995378e-03, -1.95959304e-03],
      dtype=float32)
```

creating Embedding Matrix for Vectors for words

In [43]:

```
word_vectors = model.wv
len(word_vectors.vocab)
```

Out[43]:

```
687
```

In [0]:

```
embedding_matrix = np.zeros((len(word_vectors.vocab), 300))
for word, i in t.word_index.items():
  if word in word_vectors.vocab and word in t.word_index.keys():
    embedding_vector = model[word]
    if embedding_vector is not None:
```

```
        embedding_matrix[i] = embedding_vector
```

Creating Model for Words

Model for Non-Contextual-vectors without a Embedding Matrix for Words

Intialize Vectors for Each word with random vectors and Learn those Vectors for Words

In [0]:

```
from keras.layers import Dense, Dropout, Embedding, LSTM, Bidirectional
sequence_input = Input(shape=(13,))
# Embedding Layer
embedded_sequences = Embedding(len(word_vectors.vocab),output_dim=150,input_length=13)
(sequence_input)
# LSTM  Layer using Sequence from Backward-last
# LSTM  Layer using Sequence from Forward Sequence
lstm_2_model = Bidirectional(LSTM(64,return_sequences=True,dropout=0.5))(embedded_sequences)


flatten_layer_3 = Flatten()
X_2 = flatten_layer_3(lstm_2_model)
```

In [46]:

```
X_2
```

Out[46]:

```
<tf.Tensor 'flatten_3/Reshape:0' shape=(?, ?) dtype=float32>
```

Training Bi-directional LSTM using a Contextual vectors obtained from gensim for Words

In [0]:

```
embedded_sequences_question =
Embedding(len(word_vectors.vocab),output_dim=300,weights=[embedding_matrix],input_length=13,trainab
le=False)(sequence_input)

lstm_out_model_backward_1 = Bidirectional(LSTM(64,return_sequences=True,dropout=0.5))
(embedded_sequences_question)

flatten_layer_4 = Flatten()
X = flatten_layer_4(lstm_out_model_backward_1)
```

Concatenate Output of LSTM's from Non-contextual and Contextual Vectors for words and Char's Models

In [0]:

```
numerical_concatenate_5 = keras.layers.concatenate([X_2_char,
                                                    X_char,
                                                    X_2,
                                                    X],axis=-1)
```

Train Few Dense Layers at the End

In [0]:

```
Dense_1 = Dense(64, activation='relu')(numerical_concatenate_5)
dropout_1 = Dropout(0.5)(Dense_1)
Dense_2 = Dense(32, activation='relu')(dropout_1)
dropout_2 = Dropout(0.5)(Dense_2)
```

```
Dense_3 = Dense(16, activation='relu')(dropout_2)

predictions = Dense(4, activation='softmax')(Dense_3)
```

In [0]:

```
model_1 = Model(inputs= [
                          sequence_input_char,sequence_input], outputs = predictions)
```

In [51]:

```
model_1.summary()
```

```
Model: "model_1"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            (None, 45)           0
__________________________________________________________________________________________________
input_2 (InputLayer)            (None, 13)           0
__________________________________________________________________________________________________
embedding_1 (Embedding)         (None, 45, 150)      47550       input_1[0][0]
__________________________________________________________________________________________________
embedding_2 (Embedding)         (None, 45, 300)      95100       input_1[0][0]
__________________________________________________________________________________________________
embedding_3 (Embedding)         (None, 13, 150)      103050      input_2[0][0]
__________________________________________________________________________________________________
embedding_4 (Embedding)         (None, 13, 300)      206100      input_2[0][0]
__________________________________________________________________________________________________
bidirectional_1 (Bidirectional) (None, 45, 128)      110080      embedding_1[0][0]
__________________________________________________________________________________________________
bidirectional_2 (Bidirectional) (None, 45, 128)      186880      embedding_2[0][0]
__________________________________________________________________________________________________
bidirectional_3 (Bidirectional) (None, 13, 128)      110080      embedding_3[0][0]
__________________________________________________________________________________________________
bidirectional_4 (Bidirectional) (None, 13, 128)      186880      embedding_4[0][0]
__________________________________________________________________________________________________
flatten_1 (Flatten)             (None, 5760)         0           bidirectional_1[0][0]
__________________________________________________________________________________________________
flatten_2 (Flatten)             (None, 5760)         0           bidirectional_2[0][0]
__________________________________________________________________________________________________
flatten_3 (Flatten)             (None, 1664)         0           bidirectional_3[0][0]
__________________________________________________________________________________________________
flatten_4 (Flatten)             (None, 1664)         0           bidirectional_4[0][0]
__________________________________________________________________________________________________
concatenate_1 (Concatenate)     (None, 14848)        0           flatten_1[0][0]
                                                                 flatten_2[0][0]
                                                                 flatten_3[0][0]
                                                                 flatten_4[0][0]
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, 64)           950336      concatenate_1[0][0]
__________________________________________________________________________________________________
dropout_1 (Dropout)             (None, 64)           0           dense_1[0][0]
__________________________________________________________________________________________________
dense_2 (Dense)                 (None, 32)           2080        dropout_1[0][0]
__________________________________________________________________________________________________
dropout_2 (Dropout)             (None, 32)           0           dense_2[0][0]
__________________________________________________________________________________________________
dense_3 (Dense)                 (None, 16)           528         dropout_2[0][0]
__________________________________________________________________________________________________
dense_4 (Dense)                 (None, 4)            68          dense_3[0][0]
==================================================================================================
Total params: 1,998,732
Trainable params: 1,697,532
Non-trainable params: 301,200
__________________________________________________________________________________________________
```

Store the Best Model while training which has the highest Val-Accuracy

In [52]:

```
from sklearn.metrics import roc_curve, auc
```

```
opt = Adam(lr=0.01, beta_1=0.9, beta_2=0.999, decay=0.01)
model_1.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/optimizers.py:793: The name t
f.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:3576: The name tf.log is deprecated. Please use tf.ma
th.log instead.
```

In [0]:

```
# creating a model checkpoint which monitirs the training Loss
from keras.callbacks import ModelCheckpoint
# Model stores the Parameters of Best Model which has low training-Loss
filepath = "/content/drive/My Drive/quadratyx/weights_full_train-{epoch:02d}-{val_acc:.2f}.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='val_acc', verbose=1, save_best_only=True, mode='max
')
callbacks_list = [checkpoint]
```

In [54]:

```
history = model_1.fit([padded_docs_train_char,padded_docs_train],y_encoded[:400],epochs = 50,batch
_size=32,
                      validation_data=([padded_docs_test_char,padded_docs_test],y_encoded[400:]),ca
llbacks=callbacks_list)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/tensorflow_core/python/ops/math_grad.py:1424: where (from
tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:1033: The name tf.assign_add is deprecated. Please us
e tf.compat.v1.assign_add instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:1020: The name tf.assign is deprecated. Please use tf
.compat.v1.assign instead.

Train on 400 samples, validate on 113 samples
Epoch 1/50
400/400 [==============================] - 12s 29ms/step - loss: 1.4111 - acc: 0.2675 - val_loss:
1.3734 - val_acc: 0.3451

Epoch 00001: val_acc improved from -inf to 0.34513, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-01-0.35.hdf5
Epoch 2/50
400/400 [==============================] - 4s 10ms/step - loss: 1.3988 - acc: 0.2400 - val_loss: 1
.3722 - val_acc: 0.3186

Epoch 00002: val_acc did not improve from 0.34513
Epoch 3/50
400/400 [==============================] - 4s 10ms/step - loss: 1.3886 - acc: 0.2875 - val_loss: 1
.3611 - val_acc: 0.3274

Epoch 00003: val_acc did not improve from 0.34513
Epoch 4/50
400/400 [==============================] - 4s 10ms/step - loss: 1.3794 - acc: 0.2875 - val_loss: 1
.3629 - val_acc: 0.3186

Epoch 00004: val_acc did not improve from 0.34513
Epoch 5/50
400/400 [==============================] - 4s 10ms/step - loss: 1.3795 - acc: 0.2800 - val_loss: 1
.3392 - val_acc: 0.3540

Epoch 00005: val_acc improved from 0.34513 to 0.35398, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-05-0.35.hdf5
Epoch 6/50
400/400 [==============================] - 4s 10ms/step - loss: 1.3619 - acc: 0.3275 - val_loss: 1
.2963 - val_acc: 0.4513

Epoch 00006: val_acc improved from 0.35398 to 0.45133, saving model to /content/drive/My
```

```
Epoch 00006: val_acc improved from 0.39590 to 0.45133, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-06-0.45.hdf5
Epoch 7/50
400/400 [==============================] - 4s 10ms/step - loss: 1.2865 - acc: 0.4025 - val_loss: 1
.1708 - val_acc: 0.4867

Epoch 00007: val_acc improved from 0.45133 to 0.48673, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-07-0.49.hdf5
Epoch 8/50
400/400 [==============================] - 4s 10ms/step - loss: 1.1761 - acc: 0.4325 - val_loss: 1
.0633 - val_acc: 0.4248

Epoch 00008: val_acc did not improve from 0.48673
Epoch 9/50
400/400 [==============================] - 4s 10ms/step - loss: 1.1179 - acc: 0.4575 - val_loss: 1
.0534 - val_acc: 0.5221

Epoch 00009: val_acc improved from 0.48673 to 0.52212, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-09-0.52.hdf5
Epoch 10/50
400/400 [==============================] - 4s 10ms/step - loss: 1.1134 - acc: 0.4400 - val_loss: 1
.0742 - val_acc: 0.4513

Epoch 00010: val_acc did not improve from 0.52212
Epoch 11/50
400/400 [==============================] - 4s 10ms/step - loss: 1.0087 - acc: 0.4850 - val_loss: 1
.0735 - val_acc: 0.5044

Epoch 00011: val_acc did not improve from 0.52212
Epoch 12/50
400/400 [==============================] - 4s 10ms/step - loss: 1.0104 - acc: 0.4950 - val_loss: 1
.1214 - val_acc: 0.5044

Epoch 00012: val_acc did not improve from 0.52212
Epoch 13/50
400/400 [==============================] - 4s 10ms/step - loss: 0.9701 - acc: 0.5100 - val_loss: 1
.1438 - val_acc: 0.5133

Epoch 00013: val_acc did not improve from 0.52212
Epoch 14/50
400/400 [==============================] - 4s 10ms/step - loss: 0.9429 - acc: 0.4775 - val_loss: 1
.2749 - val_acc: 0.5221

Epoch 00014: val_acc improved from 0.52212 to 0.52212, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-14-0.52.hdf5
Epoch 15/50
400/400 [==============================] - 4s 10ms/step - loss: 0.8979 - acc: 0.5300 - val_loss: 1
.2471 - val_acc: 0.5133

Epoch 00015: val_acc did not improve from 0.52212
Epoch 16/50
400/400 [==============================] - 4s 10ms/step - loss: 0.8923 - acc: 0.5100 - val_loss: 1
.3979 - val_acc: 0.5310

Epoch 00016: val_acc improved from 0.52212 to 0.53097, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-16-0.53.hdf5
Epoch 17/50
400/400 [==============================] - 4s 10ms/step - loss: 0.8787 - acc: 0.5175 - val_loss: 1
.4580 - val_acc: 0.5133

Epoch 00017: val_acc did not improve from 0.53097
Epoch 18/50
400/400 [==============================] - 4s 10ms/step - loss: 0.9191 - acc: 0.4975 - val_loss: 1
.2199 - val_acc: 0.5044

Epoch 00018: val_acc did not improve from 0.53097
Epoch 19/50
400/400 [==============================] - 4s 10ms/step - loss: 0.8725 - acc: 0.5450 - val_loss: 1
.4513 - val_acc: 0.4956

Epoch 00019: val_acc did not improve from 0.53097
Epoch 20/50
400/400 [==============================] - 4s 10ms/step - loss: 0.8539 - acc: 0.5350 - val_loss: 1
.4730 - val_acc: 0.5044

Epoch 00020: val_acc did not improve from 0.53097
Epoch 21/50
400/400 [==============================] - 4s 10ms/step - loss: 0.8414 - acc: 0.5325 - val_loss: 1
```

```
400/400 [==============================] - 4s 10ms/step - loss: 0.6414 - acc: 0.5525 - val_loss: 1
.4484 - val_acc: 0.5221

Epoch 00021: val_acc did not improve from 0.53097
Epoch 22/50
400/400 [==============================] - 4s 10ms/step - loss: 0.8312 - acc: 0.5425 - val_loss: 1
.4279 - val_acc: 0.5310

Epoch 00022: val_acc improved from 0.53097 to 0.53097, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-22-0.53.hdf5
Epoch 23/50
400/400 [==============================] - 4s 10ms/step - loss: 0.8276 - acc: 0.5700 - val_loss: 1
.4532 - val_acc: 0.5133

Epoch 00023: val_acc did not improve from 0.53097
Epoch 24/50
400/400 [==============================] - 4s 10ms/step - loss: 0.7864 - acc: 0.5800 - val_loss: 1
.4495 - val_acc: 0.5310

Epoch 00024: val_acc did not improve from 0.53097
Epoch 25/50
400/400 [==============================] - 4s 10ms/step - loss: 0.7719 - acc: 0.6125 - val_loss: 1
.6396 - val_acc: 0.5398

Epoch 00025: val_acc improved from 0.53097 to 0.53982, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-25-0.54.hdf5
Epoch 26/50
400/400 [==============================] - 4s 10ms/step - loss: 0.7553 - acc: 0.6100 - val_loss: 1
.6023 - val_acc: 0.5841

Epoch 00026: val_acc improved from 0.53982 to 0.58407, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-26-0.58.hdf5
Epoch 27/50
400/400 [==============================] - 4s 10ms/step - loss: 0.7829 - acc: 0.6550 - val_loss: 1
.1800 - val_acc: 0.5664

Epoch 00027: val_acc did not improve from 0.58407
Epoch 28/50
400/400 [==============================] - 4s 10ms/step - loss: 0.6753 - acc: 0.6850 - val_loss: 1
.5403 - val_acc: 0.5310

Epoch 00028: val_acc did not improve from 0.58407
Epoch 29/50
400/400 [==============================] - 4s 10ms/step - loss: 0.6786 - acc: 0.6975 - val_loss: 1
.4073 - val_acc: 0.5575

Epoch 00029: val_acc did not improve from 0.58407
Epoch 30/50
400/400 [==============================] - 4s 10ms/step - loss: 0.6239 - acc: 0.7050 - val_loss: 1
.4370 - val_acc: 0.6460

Epoch 00030: val_acc improved from 0.58407 to 0.64602, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-30-0.65.hdf5
Epoch 31/50
400/400 [==============================] - 4s 10ms/step - loss: 0.6256 - acc: 0.7025 - val_loss: 1
.3952 - val_acc: 0.5841

Epoch 00031: val_acc did not improve from 0.64602
Epoch 32/50
400/400 [==============================] - 4s 10ms/step - loss: 0.6428 - acc: 0.7325 - val_loss: 1
.4388 - val_acc: 0.5575

Epoch 00032: val_acc did not improve from 0.64602
Epoch 33/50
400/400 [==============================] - 4s 10ms/step - loss: 0.5218 - acc: 0.7300 - val_loss: 1
.5496 - val_acc: 0.5575

Epoch 00033: val_acc did not improve from 0.64602
Epoch 34/50
400/400 [==============================] - 4s 10ms/step - loss: 0.5385 - acc: 0.7575 - val_loss: 1
.6802 - val_acc: 0.6637

Epoch 00034: val_acc improved from 0.64602 to 0.66372, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-34-0.66.hdf5
Epoch 35/50
400/400 [==============================] - 4s 10ms/step - loss: 0.5154 - acc: 0.7975 - val_loss: 1
.6203 - val_acc: 0.6018
```

```
Epoch 00035: val_acc did not improve from 0.66372
Epoch 36/50
400/400 [==============================] - 4s 10ms/step - loss: 0.4669 - acc: 0.8050 - val_loss: 1
.8894 - val_acc: 0.6637

Epoch 00036: val_acc did not improve from 0.66372
Epoch 37/50
400/400 [==============================] - 4s 10ms/step - loss: 0.3579 - acc: 0.8425 - val_loss: 2
.0913 - val_acc: 0.6549

Epoch 00037: val_acc did not improve from 0.66372
Epoch 38/50
400/400 [==============================] - 4s 10ms/step - loss: 0.3714 - acc: 0.8550 - val_loss: 2
.0690 - val_acc: 0.6460

Epoch 00038: val_acc did not improve from 0.66372
Epoch 39/50
400/400 [==============================] - 4s 10ms/step - loss: 0.4010 - acc: 0.8450 - val_loss: 1
.8371 - val_acc: 0.6460

Epoch 00039: val_acc did not improve from 0.66372
Epoch 40/50
400/400 [==============================] - 4s 10ms/step - loss: 0.3367 - acc: 0.8675 - val_loss: 1
.9505 - val_acc: 0.6814

Epoch 00040: val_acc improved from 0.66372 to 0.68142, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-40-0.68.hdf5
Epoch 41/50
400/400 [==============================] - 4s 10ms/step - loss: 0.2992 - acc: 0.8825 - val_loss: 1
.9680 - val_acc: 0.6372

Epoch 00041: val_acc did not improve from 0.68142
Epoch 42/50
400/400 [==============================] - 4s 10ms/step - loss: 0.2704 - acc: 0.8725 - val_loss: 1
.9501 - val_acc: 0.6637

Epoch 00042: val_acc did not improve from 0.68142
Epoch 43/50
400/400 [==============================] - 4s 10ms/step - loss: 0.2131 - acc: 0.9175 - val_loss: 2
.1507 - val_acc: 0.6372

Epoch 00043: val_acc did not improve from 0.68142
Epoch 44/50
400/400 [==============================] - 4s 10ms/step - loss: 0.1930 - acc: 0.9375 - val_loss: 2
.0675 - val_acc: 0.7080

Epoch 00044: val_acc improved from 0.68142 to 0.70796, saving model to /content/drive/My
Drive/quadratyx/weights_full_train-44-0.71.hdf5
Epoch 45/50
400/400 [==============================] - 4s 9ms/step - loss: 0.1791 - acc: 0.9275 - val_loss: 2.
3499 - val_acc: 0.6549

Epoch 00045: val_acc did not improve from 0.70796
Epoch 46/50
400/400 [==============================] - 4s 10ms/step - loss: 0.1790 - acc: 0.9300 - val_loss: 2
.1180 - val_acc: 0.6814

Epoch 00046: val_acc did not improve from 0.70796
Epoch 47/50
400/400 [==============================] - 4s 10ms/step - loss: 0.1805 - acc: 0.9225 - val_loss: 2
.0886 - val_acc: 0.6814

Epoch 00047: val_acc did not improve from 0.70796
Epoch 48/50
400/400 [==============================] - 4s 10ms/step - loss: 0.1804 - acc: 0.9425 - val_loss: 2
.1005 - val_acc: 0.6549

Epoch 00048: val_acc did not improve from 0.70796
Epoch 49/50
400/400 [==============================] - 4s 9ms/step - loss: 0.1901 - acc: 0.9350 - val_loss: 2.
1148 - val_acc: 0.6460

Epoch 00049: val_acc did not improve from 0.70796
Epoch 50/50
400/400 [==============================] - 4s 10ms/step - loss: 0.1303 - acc: 0.9550 - val_loss: 2
.1249 - val_acc: 0.6726
```

```
Epoch 00050: val_acc did not improve from 0.70796
```

Model Diagram

In [55]:

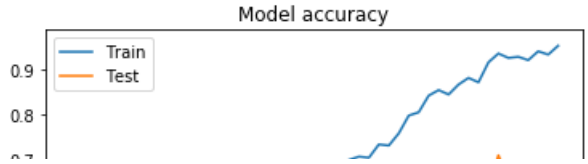
```
from keras.utils import plot_model
plot_model(model_1, to_file='model.png')
```

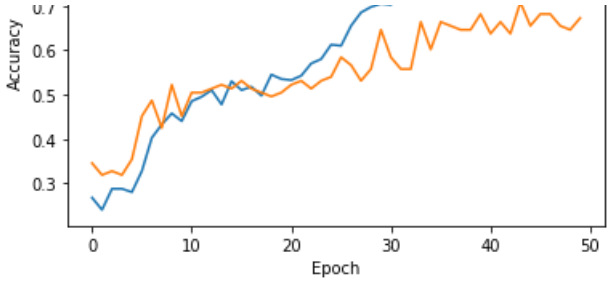
Out[55]:

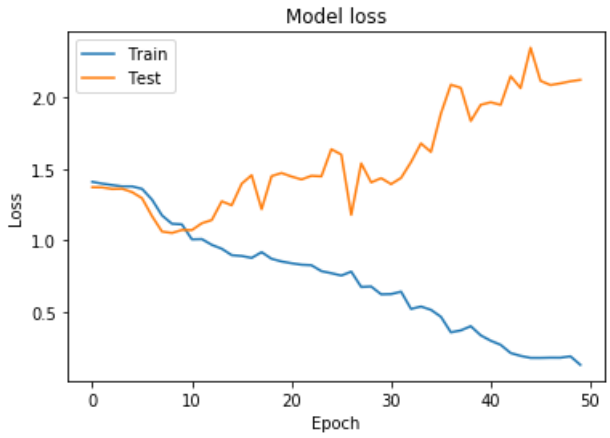
Model's plot's Train and Valid-ACC

In [56]:

```
# Plot training & validation accuracy values
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()

# Plot training & validation loss values
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```







Load the Best Model

In [0]:

```
from keras.models import load_model

model_1 = load_model('/content/drive/My Drive/quadratyx/weights_full_train-44-0.71.hdf5')
```

train Confusion matrix, Precision Matrix, Recall Matrix

In [0]:

```
y_predicted_train = model_1.predict([padded_docs_train_char,padded_docs_train])
```

In [0]:

```
y_predicted_label= [np.argmax(y_predicted_train[i])+1 for i in range(y_predicted_train.shape[0])]
```

In [0]:

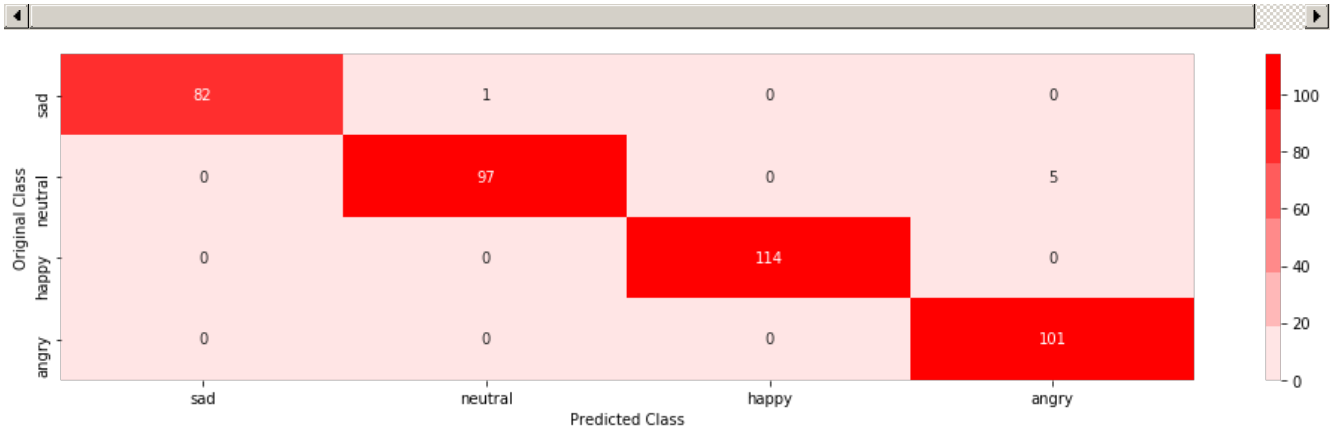
```
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
import seaborn as sns
def plot_confusion_matrix(test_y, predict_y):
    C = confusion_matrix(test_y, predict_y)

    # C = 17,17 matrix, each cell (i,j) represents number of points of class i are predicted class
j

    A =(((C.T)/(C.sum(axis=1))).T)
    #divid each element of the confusion matrix with the sum of elements in that column

    # C = [[1, 2],
    #     [3, 4]]
    # C.T = [[1, 3],
    #        [2, 4]]
    # C.sum(axis = 1)  axis=0 corresonds to columns and axis=1 corresponds to rows in two
diamensional array
    # C.sum(axix =1) = [[3, 7]]
    # ((C.T)/(C.sum(axis=1))) = [[1/3, 3/7]
    #                           [2/3, 4/7]]

    # ((C.T)/(C.sum(axis=1))).T = [[1/3, 2/3]
```

```
    #                            [3/7, 4/7]]
    # sum of row elements = 1

    B =(C/C.sum(axis=0))
    #divid each element of the confusion matrix with the sum of elements in that row
    # C = [[1, 2],
    #      [3, 4]]
    # C.sum(axis = 0)  axis=0 corresonds to columns and axis=1 corresponds to rows in two
diamensional array
    # C.sum(axix =0) = [[4, 6]]
    # (C/C.sum(axis=0)) = [[1/4, 2/6],
    #                      [3/4, 4/6]]
    # Confusion matrix
    labels = ['sad','neutral','happy','angry']
    cmap=sns.light_palette("red")
    # representing A in heatmap format
    print("-"*50, "Confusion matrix", "-"*50)
    plt.figure(figsize=(17,4))
    sns.heatmap(C, annot=True, cmap=cmap, fmt='d',xticklabels=labels, yticklabels=labels)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.show()

    # PRECISION MATRIX
    print("-"*50, "Precision matrix", "-"*50)
    cmap=sns.light_palette("green")
    plt.figure(figsize=(17,4))
    sns.heatmap(B, annot=True, cmap=cmap, fmt=".2g", xticklabels=labels, yticklabels=labels)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.show()
    print("Sum of columns in precision matrix",B.sum(axis=0))

    # RECALL MATRIX
    # representing B in heatmap format
    print("-"*50, "Recall matrix"    , "-"*50)
    cmap=sns.light_palette("blue")
    plt.figure(figsize=(17,4))
    sns.heatmap(A, annot=True, cmap=cmap, fmt=".2g", xticklabels=labels, yticklabels=labels)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.show()
    print("Sum of rows in precision matrix",A.sum(axis=1))
    # how many images are wrongly Predicted
    print("Number of misclassified points ",(len(test_y)-np.trace(C))/len(test_y)*100)
```

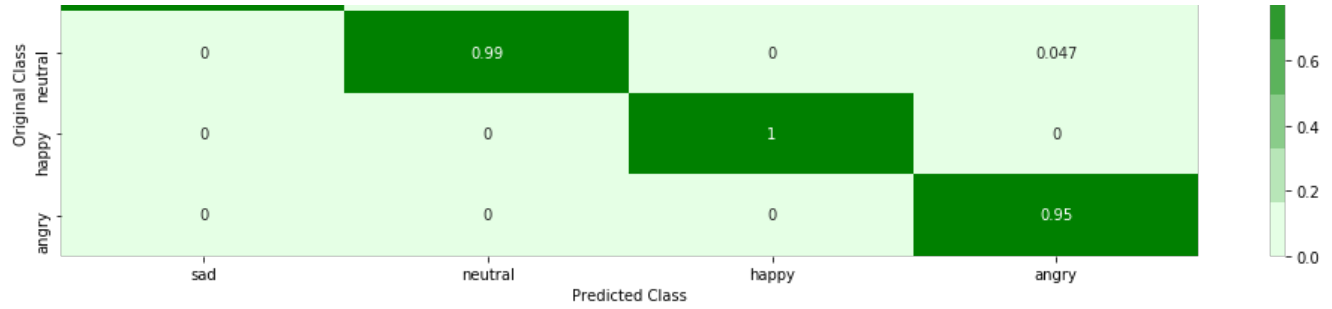
In [62]:

```
plot_confusion_matrix(output_y[:400],y_predicted_label)
```

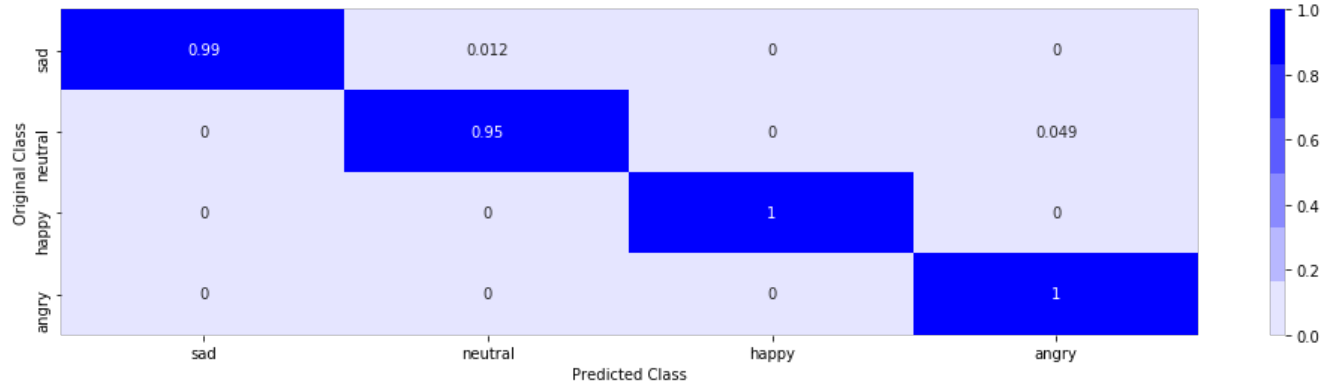
```
---------------------------------------------------- Confusion matrix ----------------------------------
----------------
```




```
---------------------------------------------------- Precision matrix ----------------------------------
----------------
```




```
Sum of columns in precision matrix [1. 1. 1. 1.]
-------------------------------------------------- Recall matrix ----------------------------------
-------------
```




```
Sum of rows in precision matrix [1. 1. 1. 1.]
Number of misclassified points  1.5
```

Test Confusion Matrix, PrecisionMatrix,Recall Matrix

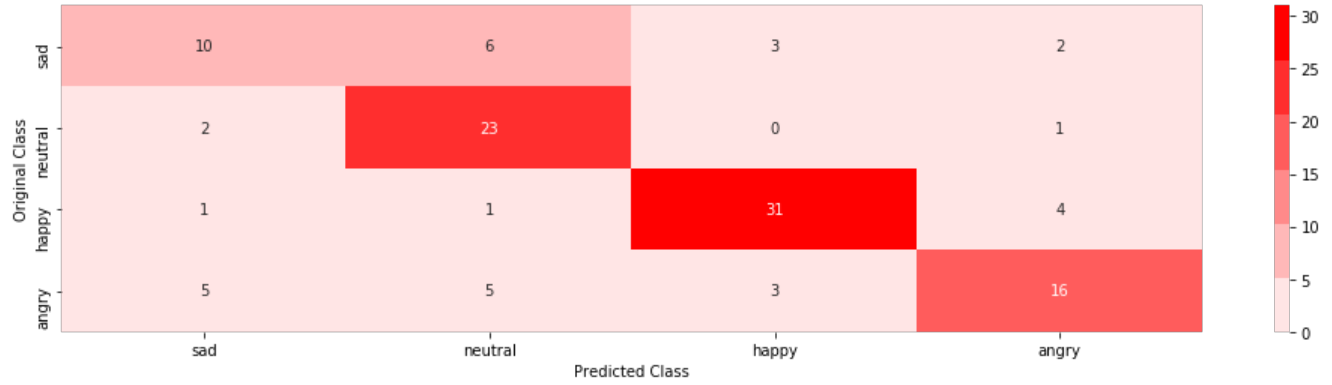
In [0]:

```
y_predicted_test = model_1.predict([padded_docs_test_char,padded_docs_test])

y_predicted_label= [np.argmax(y_predicted_test[i])+1 for i in range(y_predicted_test.shape[0])]
```

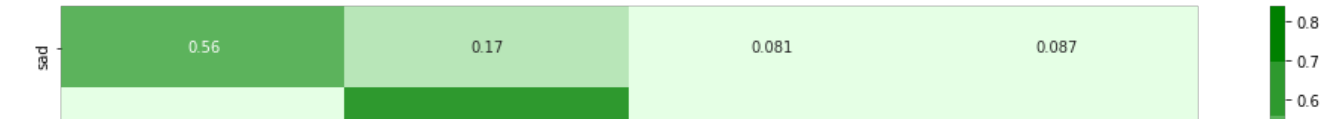
In [64]:

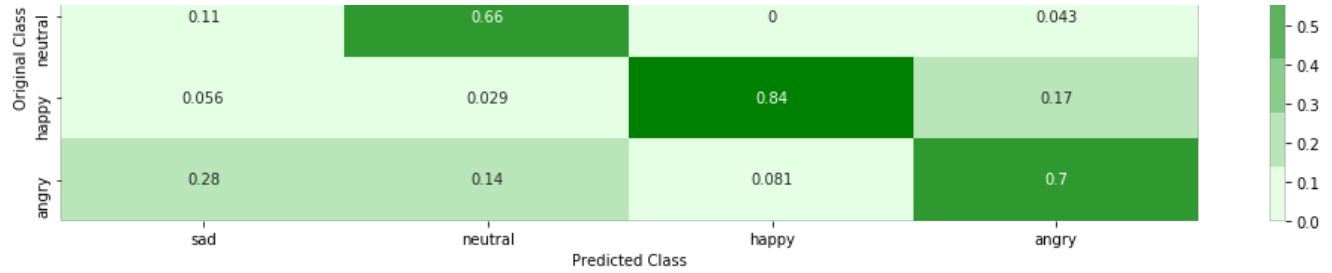
```
plot_confusion_matrix(output_y[400:],y_predicted_label)
```

```
-------------------------------------------------- Confusion matrix -------------------------------
----------------
```

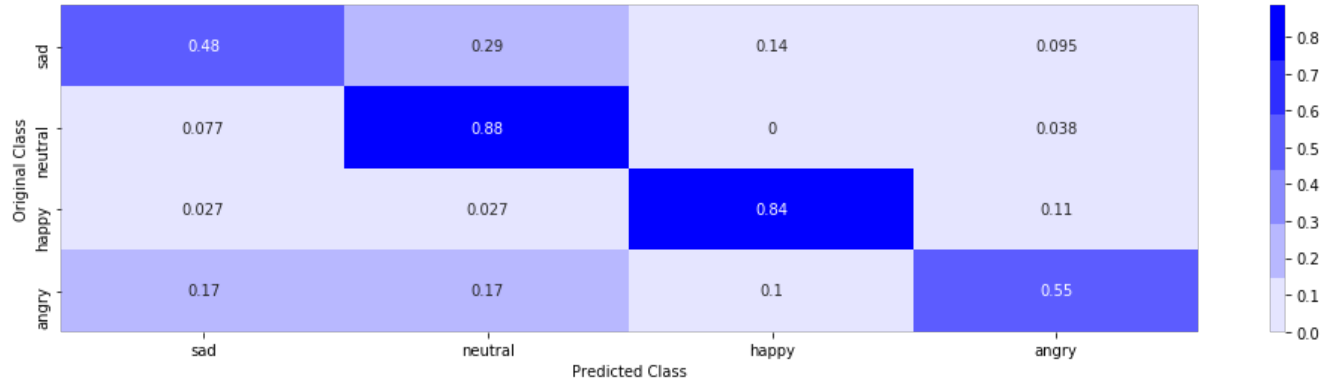



```
-------------------------------------------------- Precision matrix -------------------------------
----------------
```






```
Sum of columns in precision matrix [1. 1. 1. 1.]
-------------------------------------------------- Recall matrix ------------------------------------
-------------
```




```
Sum of rows in precision matrix [1. 1. 1. 1.]
Number of misclassified points  29.20353982300885
```