Text Classification:

Data

```
1. we have total of 20 types of documents(Text files) and t
otal 18828 documents(text files).
2. You can download data from this link 
(https://drive.google.com/open?id=1rxD15nyeIPIAZ-J2VYPrDRZI
66-TBWvM), in that you will get documents.rar folder.
If you unzip that, you will get total of 18828 documnets. d
ocument name is defined as'ClassLabel_DocumentNumberInThatL
abel'.
so from document name, you can extract the label for that d
ocument.
4. Now our problem is to classify all the documents into an
y one of the class.
5. Below we provided count plot of all the labels in our da
ta.
```

In [14]:
```
### count plot of all the class labels.
```

Assignment:

In [1]:
```
import pandas as pd
import numpy as np
from collections import defaultdict
import spacy
import en_core_web_sm
nlp = en_core_web_sm.load()
import re
import warnings
warnings.filterwarnings('ignore')
```

In [15]:
```
import tensorflow as tf
from keras.models import Sequential
from tensorflow.keras.layers import Input
from keras.layers import Dense,Conv2D,MaxPool1D,Activation,Dropout,
from keras.models import Model
from keras.initializers import RandomUniform, HeUniform
from keras.optimizers import Adam
from tensorflow.keras.utils import plot_model
import nltk
```

In [16]:
```
get_ipython().system_raw("unrar x documents.rar")
```

In [17]:
```
import os

directory = r'documents'
results = defaultdict(list)
for filename in os.listdir(directory):
    with open(os.path.join(directory, filename), 'r', encoding = "I
        results['file_name'].append(filename.split('_')[1])
        results['text'].append(file.read())
        results['label'].append(filename.split('_')[0])

df = pd.DataFrame(results)
```

In [18]:
```
df
```

Out[18]:

	file_name	text	label
0	60933.txt	From: nsmca@aurora.alaska.edu\nSubject: Re: St...	sci.space
1	76278.txt	From: coates@bigwpi.WPI.EDU (Jeffery David Coa...	misc.forsale
2	102900.txt	From: dlb5404@tamuts.tamu.edu (Daryl Biberdorf...	rec.autos
3	66999.txt	From: howardy@freud.nia.nih.gov (Howard Wai-Ch...	comp.windows.x
4	53887.txt	From: cdash@moet.cs.colorado.edu (Charles Shub...	rec.sport.hockey
...	...	...	...
18823	50426.txt	From: jcav@ellis.uchicago.edu (JohnC)\nSubject...	comp.sys.mac.hardware
18824	104895.txt	From: thagerma@magnus.acs.ohio-state.edu (Tere...	rec.sport.baseball
18825	67214.txt	From: kriss@frec.bull.fr (Christian Mollard)\n...	comp.windows.x
18826	54943.txt	From: stevef@bug.UUCP (Steven R Fordyce)\nSubj...	talk.politics.guns
18827	104392.txt	From: asphaug@lpl.arizona.edu (Erik Asphaug x2...	rec.motorcycles

18828 rows × 3 columns

In [19]:
```
df['text'][0]
```

Out[19]:
```
"From: nsmca@aurora.alaska.edu\nSubject: Re: Stereo Pix of planets
?y\n\nIn article <1993Apr20.010326.8634@csus.edu>, arthurc@sfsuvax
1.sfsu.edu (Arthur Chandler) writes:\n> Can anyone tell me where I
might find stereo images of planetary and\n> planetary satellite s
urfaces?  GIFs preferred, but any will do.  I'm\n> especially inte
rested in stereos of the surfaces of Phobos, Deimos, Mars\n> and t
he Moon (in that order).\n>   Thanks. \n\n\names.arc.nasa.gov not
sure what subdirectory thou..\n\n==\nMichael Adams, nsmca@acad3.al
aska.edu -- I'm not high, just jacked\n\nPS: I know it has a GIF a
rea as well as SPACE and other info..\n\n"
```

In [20]:
```
def decontracted(phrase):
    # specific
    phrase = re.sub(r"won't", "will not", phrase)
    phrase = re.sub(r"can\'t", "can not", phrase)
    phrase = re.sub(r"let\'s", "let us", phrase)

    # general
    phrase = re.sub(r"n\'t", " not", phrase)
    phrase = re.sub(r"\'re", " are", phrase)
    phrase = re.sub(r"\'s", " is", phrase)
    phrase = re.sub(r"\'d", " would", phrase)
    phrase = re.sub(r"\'ll", " will", phrase)
    phrase = re.sub(r"\'t", " not", phrase)
    phrase = re.sub(r"\'ve", " have", phrase)
    phrase = re.sub(r"\'m", " am", phrase)
    return phrase
```

Preprocessing

In [22]:
```
def preprocess_text(input_text):

    pp_text = input_text

    ## preprocessing emails
    pp_email = ""
    emails = []
    new = []
    text = " "
    emails = re.findall('([a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-
    a1 = [item.split('@')[1] for item in emails]
    for item in a1:
      new.extend(item.split('.'))
    y = [s for s in new if (len(s) > 2 and s != 'com')]
    pp_email = text.join(y)

    ## replacing all emails with space
    temp = pp_text
    for email in emails:
      temp = temp.replace(email, ' ')

    pp_text = temp

    ## removing words starting with : in subject and updating whole
    temp = pp_text ##input_text
    lst = re.findall('Subject:.*', temp)
    temp = re.sub(r"\w*: ", '', temp)
    pp_subject = re.sub(r"\w*: ", '', lst[0])
    temp = temp.replace(pp_subject, ' ')
    pp_text = temp

    ## removing From: and Write To:
```

```
    lst = re.findall('From:.*', pp_text)
    if lst:
      pp_text = (pp_text).replace(lst[0], '')
    lst = re.findall('Write To:.*', pp_text)
    if lst:
      pp_text = (pp_text).replace(lst[0], '')


    ## removing characters and other conditions
    pp_text = re.sub("[<>]", " ", pp_text)  # < > removal
    pp_text = re.sub(r'\([^()]*\)', '', pp_text)  # brackets remova
    pp_text = pp_text.replace('\t', ' ').replace('\n',' ').replace(
    pp_text = re.sub(r'\w+:', '', pp_text) # removal of words endin
    pp_text = pp_text.lower()  # lowercase all letters
    lst = re.findall(r"(\w+_\b)", pp_text)  # removal of _ in suffi
    for word in lst:
        temp = word.replace('_','')
        pp_text = pp_text.replace(word, temp)

    lst = re.findall(r"(\b_\w+)", pp_text)  # removal of _ in suffi
    for word in lst:
        temp = word.replace('_','')
        pp_text = pp_text.replace(word, temp)
    pp_text = decontracted(pp_text)  # Decontractions


    ## chunking of sentences; remove person and add _ in other phra
    my_sent = pp_text
    doc = nlp(my_sent)
    dict_of_phrases = {X.text:X.label_ for X in doc.ents}

    for name, label in dict_of_phrases.items():
        if label.lower() == 'person':
            my_sent = my_sent.replace(name, '')
        if label.lower() != 'person':
            temp = name.replace(' ', '_')
            my_sent = my_sent.replace(name, temp)
    pp_text = my_sent


    pp_text = re.sub(r"\d", "", pp_text)  # delete all digits
    pp_text = re.sub(r"[0-9a-zA-Z_]?[0-9a-zA-Z_]_", "", pp_text)  #
    pp_text = re.sub(r'\b\w{15,}\b', '', pp_text)  # removal of wor
    pp_text = re.sub(r'\b\w{,2}\b', '', pp_text)
    pp_text = re.sub(r"[^a-zA-Z_ ]", "", pp_text)  # remove all wor
    pp_text = re.sub(r"[ \t]+", " ", pp_text)  # trim extra spaces

    return pp_email, pp_subject, pp_text
```

In [23]:
```
text = df['text'][500]
pp_email, pp_subject, pp_text = preprocess_text(text)
```

In [24]:
```
print(pp_email)
print(pp_subject)
print(pp_text)
print(df['label'][500])
```

```
cmu edu
"Illegal" tint windows
 know long shot but maybe someone went through this and will have
some comments share the story bought car out state and trying get
the safety inspection pennsylvania the problem that the car has af
termarket tint all windows except the windshield the tint rather w
eak and you can clearly see the inside the car through the tint th
e inspection garage said that they will not pass unless get waiver
from the state police went the state police the officer told that
aftermarket tint illegal and can get waiver only for pre car for m
edical reason asked him show the section the vehicle code that say
s illegal showed and the paraghaph said that you can not have tint
you can not see the inside the car because the tint when told him
that you can fact see the inside very well shut the book and said
just illegal and fact can have someone give you ticket for right n
ow well will not argue with that since the vehicle code says long
you can see through the tint would like keep would also like get s
ome sort paper from the police that says can get the inspection an
d that will not get trouble for the tint later also would not mind
registering complaint against that officer really pissed off does
anyone have any experience getting that sort paper from the police
especially pennsylvania does anyone have any experience registerin
g complaint against officer called the station later today but the
y basically said there place where could register complaint agains
t officer and decide keep the tint and get ticket anyway how much
chance stand succesfully appeal the ticket court any comments abou
t will welcome michal
rec.autos
```

In [25]:
```
data = pd.DataFrame(index=range(len(df)), columns=['preprocessed_em
for i in range(len(df)):
    data['preprocessed_email'][i], data['preprocessed_subject'][i],
```

In [26]:
```
data
```

Out[26]:

	preprocessed_email	preprocessed_subject	preprocessed_text
0	aurora alaska edu csus edu sfsuvax1 sfsu edu a...	Stereo Pix of planets?y	article can anyone tell where might find ster...
1	bigwpi WPI EDU wpi wpi edu	Sony Amplifier and Crossover for sale	for sale sony car stereo amplifier rated powe...
2	tamuts tamu edu walter bellcore ctt bellcore t...	LH Workmanship	article just visited the auto show and saw tw...
3	freud nia nih gov	need shading program example in X	anyone know about any shading program based x...
4	moet colorado edu Colorado EDU uiowa edu	Thumbs WAY WAY WAY DOWN to ESPN	tuesday and the islescaps game going into ove...
...	...	...	...
18823	ellis uchicago edu midway uchicago edu uchfm b...	how do you like the Apple Color OneScanner?	are all set buy one these for the office use ...
18824	magnus acs ohio-state edu	Chicago visit	planning weekend chicago nemonth for first li...
18825	frec bull nynexst nynexst mchp sni ap542 uucp ...	Looking For David E. Smyth	article the author wcl his the only name foun...
18826	bug UUCP freenet carleton Freenet carleton	how do we stop people with a gun?	article come gun kills people rather people k...
18827	lpl arizona edu rtsg mot rtsg mot bnr bnr hind...	CAMPING was Help with backpack	article article farra crafty girfriend makes ...

18828 rows × 3 columns

Final Preprocessed Data

In [36]:
```
data['text'] = df['text']
data['class'] = df['label']
```

In [37]:
```
data.iloc[5000]
```

Out[37]:
```
preprocessed_email       Fedex Msfc Nasa Gov embl-heidelberg EMBL-H
eide...
preprocessed_subject                                          HyperKn
owledge
preprocessed_text       wingo article writes the project for nexts
tep ...
text                    From: wingo%cspara.decnet@Fedex.Msfc.Nasa.
Gov\...
class                                                              sc
i.space
Name: 5000, dtype: object
```

In [38]:
```
data["preprocessed_combined"] = data[['preprocessed_text', 'preproc
```

In [39]:
```
data["preprocessed_combined"][100]
```

Out[39]:
```
' article having problem configuring the mouse windows use com wit
h irq com and com are being used support two _hour bbs lines there
you com and com use the same irq therefore you can not use mouse c
om and modem com vice versa limitation dos and fact windows will n
ot see mouse anything other than com com accept this fact and eith
er get bus mouse get new computer would also like know possible us
e the mouse ports other than com com the advice above applies  Mou
se on Com3\x08\x08\x08OM3 or COM4 in Windows helium gas uug arizon
a edu bcstec boeing bcstec boeing ulowell edu gas uug arizona edu'
```

In [40]:
```
data["class"] = data["class"].astype('category')
data.dtypes
data["class_cat"] = data["class"].cat.codes
data.head()
```

Out[40]:

	preprocessed_email	preprocessed_subject	preprocessed_text	
0	aurora alaska edu csus edu sfsuvax1 sfsu edu a...	Stereo Pix of planets?y	article can anyone tell where might find ster...	nsmca@aurora.alaska.edu\nS F
1	bigwpi WPI EDU wpi wpi edu	Sony Amplifier and Crossover for sale	for sale sony car stereo amplifier rated powe...	From: coates@bigwpi.W (Jeffery Davic
2	tamuts tamu edu walter bellcore ctt bellcore t...	LH Workmanship	article just visited the auto show and saw tw...	From: dlb5404@tamuts.ta (Daryl Bibe
3	freud nia nih gov	need shading program example in X	anyone know about any shading program based x...	From: howardy@freud.nia. (Howard W
4	moet colorado edu Colorado EDU uiowa edu	Thumbs WAY WAY WAY DOWN to ESPN	tuesday and the islescaps game going into ove...	From: cdash@moet.cs.colora (Charles

In [49]:
```
data.to_csv('/content/final_data.csv', header=True)
```

Final Data

In [4]:
```
final_data = pd.read_csv('final_data.csv')
```

In [6]:
```
final_data = data.drop(['Unnamed: 0', 'Unnamed: 0.1'], axis=1)
```

In [7]:
```
final_data.head()
```

Out[7]:

	preprocessed_email	preprocessed_subject	preprocessed_text	
0	aurora alaska edu csus edu sfsuvax1 sfsu edu a...	Stereo Pix of planets?y	article can anyone tell where might find ster...	nsmca@aurora.alaska.edu\nS F
1	bigwpi WPI EDU wpi wpi edu	Sony Amplifier and Crossover for sale	for sale sony car stereo amplifier rated powe...	From: coates@bigwpi.W (Jeffery David
2	tamuts tamu edu walter bellcore ctt bellcore t...	LH Workmanship	article just visited the auto show and saw tw...	From: dlb5404@tamuts.ta (Daryl Bibe
3	freud nia nih gov	need shading program example in X	anyone know about any shading program based x...	From: howardy@freud.nia. (Howard W
4	moet colorado edu Colorado EDU uiowa edu	Thumbs WAY WAY WAY DOWN to ESPN	tuesday and the islescaps game going into ove...	From: cdash@moet.cs.colora (Charles

In [8]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(final_data['pre
```

In [9]:
```
print(len(X_train), len(Y_train))
print(len(X_test), len(Y_test))
```

```
14121 14121
4707 4707
```

In [10]:
```
print(X_train.shape)
print(Y_train.shape)
print(X_test.shape)
print(Y_test.shape)
```

```
(14121,)
(14121,)
(4707,)
(4707,)
```

In [13]:
```
## if there is multiclass classification, the shape of Y must be (d
y_train_encoded = tf.one_hot(Y_train, depth=20)
y_test_encoded = tf.one_hot(Y_test, depth=20)
```

In [22]:
```
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

Out[22]:
```
True
```

In [23]:
```
## referred https://www.kaggle.com/rajmehra03/a-detailed-explanatio
maxlen=-1
docs = X_train
for doc in docs:
    tokens=nltk.word_tokenize(doc)
    if(maxlen<len(tokens)):
        maxlen=len(tokens)

maxlen = max([len(s.split()) for s in final_data['preprocessed_comb
print("The maximum number of words in any document is : ",maxlen)
```

```
The maximum number of words in any document is :  8765
```

In [24]:
```
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences

maxlen = 300
docs = X_train
t1 = Tokenizer(filters='!"#$%&()*+,-./:;<=>?@[\\]^`{|}~\t\n')
t1.fit_on_texts(docs)
vocab_size1 = len(t1.word_index) + 1
encoded_docs1 = t1.texts_to_sequences(docs)
padded_docs1 = pad_sequences(encoded_docs1, maxlen, padding='post')
print(vocab_size1)
```

```
98621
```

In [25]:
```
print(len(padded_docs1))
```

```
14121
```

In [26]:
```
!wget --header="Host: downloads.cs.stanford.edu" --header="User-Age
```

```
--2021-03-12 16:16:51--  http://downloads.cs.stanford.edu/nlp/data
/glove.6B.zip
(http://downloads.cs.stanford.edu/nlp/data/glove.6B.zip)
Resolving downloads.cs.stanford.edu (downloads.cs.stanford.edu)...
171.64.64.22
Connecting to downloads.cs.stanford.edu (downloads.cs.stanford.edu
)|171.64.64.22|:80... connected.
HTTP request sent, awaiting response... 206 Partial Content
Length: 862182613 (822M), 475600333 (454M) remaining [application/
zip]
Saving to: ‘CurlWget7’

CurlWget7           100%[+++++++++===========>] 822.24M  5.06MB/s
in 86s

2021-03-12 16:18:17 (5.27 MB/s) - ‘CurlWget7’ saved [862182613/862
182613]
```

In [27]:
```
!unzip CurlWget7
```

```
Archive:  CurlWget7
  inflating: glove.6B.50d.txt
  inflating: glove.6B.100d.txt
  inflating: glove.6B.200d.txt
  inflating: glove.6B.300d.txt
```

In [28]:
```
embeddings_index = dict()
f = open('glove.6B.100d.txt')
for line in f:
    values = line.split()
    word = values[0]
    coefs = np.array(values[1:], dtype='float32')
    embeddings_index[word] = coefs
f.close()
print(len(embeddings_index))
```

```
400000
```

In [29]:
```
embedding_matrix = np.zeros((vocab_size1, 100))
for word, i in t1.word_index.items():
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector
```

In [30]:
```
vocab_size1
```

Out[30]:
```
98621
```

In [31]:
```
vocab_size_test1 = len(t1.word_index) + 1
encoded_docs_test1 = t1.texts_to_sequences(X_test)
padded_docs_test1 = pad_sequences(encoded_docs_test1, maxlen, paddi
```

Callback

In [32]:
```
class Callback(object):

    """Abstract base class used to build new callbacks.
      Attributes:
          params: dict. Training parameters
              (eg. verbosity, batch size, number of epochs...).
          model: instance of `keras.models.Model`.
              Reference of the model being trained.
          validation_data: Deprecated. Do not use.
      The `logs` dictionary that callback methods
      take as argument will contain keys for quantities relevant to
      the current batch or epoch.
      Currently, the `.fit()` method of the `Model` class
      will include the following quantities in the `logs` that
      will include the following quantities in the `logs` that
      it passes to its callbacks:
          on_epoch_end: logs include `acc` and `loss`, and
          optionally include `val_loss`
          (if validation is enabled in `fit`), and `val_acc`
          (if validation and accuracy monitoring are enabled).
          on_batch_begin: logs include `size`,
          the number of samples in the current batch.
          on_batch_end: logs include `loss`, and optionally `acc`
            (if accuracy monitoring is enabled).
      """

    def __init__(self):
        self.validation_data = None
        self.model = None
        # Whether this Callback should only run on the chief worker
        # Multi-Worker setting.
        # TODO(omalleyt): Make this attr public once solution is st
        self._chief_worker_only = None

    def set_params(self, params):
        self.params = params

    def set_model(self, model):
        self.model = model

    def on_batch_begin(self, batch, logs=None):
        """A backwards compatibility alias for `on_train_batch_begi

    def on_batch_end(self, batch, logs=None):
        """A backwards compatibility alias for `on_train_batch_end`

    def on_epoch_begin(self, epoch, logs=None):
        """Called at the start of an epoch.
        Subclasses should override for any actions to run. This fun
        be called during TRAIN mode.
        Arguments:
            epoch: integer, index of epoch.
            logs: dict. Currently no data is passed to this argumen
              but that may change in the future.
        """

    def on_epoch_end(self, epoch, logs=None):
        """Called at the end of an epoch.
        Subclasses should override for any actions to run. This fun
        be called during TRAIN mode.
        Arguments:
            epoch: integer, index of epoch.
            logs: dict, metric results for this training epoch, and
              validation epoch if validation is performed. Validati
              are prefixed with `val_`.
        """

    def on_train_batch_begin(self, batch, logs=None):
        """Called at the beginning of a training batch in `fit` met
```

```
            Subclasses should override for any actions to run.
            Arguments:
                batch: integer, index of batch within the current epoch
                logs: dict. Has keys `batch` and `size` representing th
                  number and the size of the batch.
            """
            # For backwards compatibility.
            self.on_batch_begin(batch, logs=logs)

        def on_train_batch_end(self, batch, logs=None):
            """Called at the end of a training batch in `fit` methods.
            Subclasses should override for any actions to run.
            Arguments:
                batch: integer, index of batch within the current epoch
                logs: dict. Metric results for this batch.
            """
            # For backwards compatibility.
            self.on_batch_end(batch, logs=logs)

        def on_test_batch_begin(self, batch, logs=None):
            """Called at the beginning of a batch in `evaluate` methods
            Also called at the beginning of a validation batch in the `
            methods, if validation data is provided.
            Subclasses should override for any actions to run.
            Arguments:
                batch: integer, index of batch within the current epoch
                logs: dict. Has keys `batch` and `size` representing th
                      number and the size of the batch.
            """

        def on_test_batch_end(self, batch, logs=None):
            """Called at the end of a batch in `evaluate` methods.
            Also called at the end of a validation batch in the `fit`
            methods, if validation data is provided.
            Subclasses should override for any actions to run.
            Arguments:
                batch: integer, index of batch within the current epoch
                logs: dict. Metric results for this batch.
            """

        def on_predict_batch_begin(self, batch, logs=None):
            """Called at the beginning of a batch in `predict` methods.
            Subclasses should override for any actions to run.
            Arguments:
                batch: integer, index of batch within the current epoch
                logs: dict. Has keys `batch` and `size` representing th
                      number and the size of the batch.
            """

        def on_predict_batch_end(self, batch, logs=None):
            """Called at the end of a batch in `predict` methods.
            Subclasses should override for any actions to run.
            Arguments:
```

```
            batch: integer, index of batch within the current epoch
            logs: dict. Metric results for this batch.
        """

    def on_train_begin(self, logs=None):
        """Called at the beginning of training.
        Subclasses should override for any actions to run.
        Arguments:
            logs: dict. Currently no data is passed to this argumen
                  but that may change in the future.
        """

    def on_train_end(self, logs=None):
        """Called at the end of training.
        Subclasses should override for any actions to run.
        Arguments:
            logs: dict. Currently no data is passed to this argumen
                  but that may change in the future.
        """

    def on_test_begin(self, logs=None):
        """Called at the beginning of evaluation or validation.
        Subclasses should override for any actions to run.
        Arguments:
            logs: dict. Currently no data is passed to this argumen
              but that may change in the future.
        """

    def on_test_end(self, logs=None):
        """Called at the end of evaluation or validation.
        Subclasses should override for any actions to run.
        Arguments:
            logs: dict. Currently no data is passed to this argumen
              but that may change in the future.
        """

    def on_predict_begin(self, logs=None):
        """Called at the beginning of prediction.
        Subclasses should override for any actions to run.
        Arguments:
            logs: dict. Currently no data is passed to this argumen
              but that may change in the future.
        """

    def on_predict_end(self, logs=None):
        """Called at the end of prediction.
        Subclasses should override for any actions to run.
        Arguments:
            logs: dict. Currently no data is passed to this argumen
              but that may change in the future.
    """
```

In [33]:
```
class LossHistory(tf.keras.callbacks.Callback):

    def on_train_begin(self, logs={}):
        ## on begin of training, we are creating a instance varible
        ## it is a dict with keys [loss, acc, val_loss, val_acc]
        self.val_f1s = []
        self.history={'loss': [],'acc': [],'val_loss': [],'val_acc'

    def on_epoch_end(self, epoch, logs={}):
        ## on end of each epoch, we will get logs and update the se
        self.history['loss'].append(logs.get('loss'))
        self.history['acc'].append(logs.get('acc'))
        loss = logs.get('loss')
        if logs.get('val_loss', -1) != -1:
            self.history['val_loss'].append(logs.get('val_loss'))
        if logs.get('val_acc', -1) != -1:
            self.history['val_acc'].append(logs.get('val_acc'))
        if loss is not None:
          if np.isnan(loss) or np.isinf(loss):
            print("Invalid loss and terminated at epoch {}".format(
            self.model.stop_training = True

        if logs.get('val_accuracy')>0.7:
          self.model.stop_training = True

        return
```

In [34]:
```
from keras import backend as K
def recall_m(y_true, y_pred):
    true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
    possible_positives = K.sum(K.round(K.clip(y_true, 0, 1)))
    recall = true_positives / (possible_positives + K.epsilon())
    return recall
def precision_m(y_true, y_pred):
    true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
    predicted_positives = K.sum(K.round(K.clip(y_pred, 0, 1)))
    precision = true_positives / (predicted_positives + K.epsilon()
    return precision
def f1_m(y_true, y_pred):
    precision = precision_m(y_true, y_pred)
    recall = recall_m(y_true, y_pred)
    return 2*((precision*recall)/(precision+recall+K.epsilon()))
```

Model 1

In [35]:
```
import datetime
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.metrics import accuracy_score, precision_score, recall
```

In [36]:
```
model = Sequential()
```

```
inputs = Input(shape = (maxlen,), name = 'inputs')
embed_layer = Embedding(input_dim = vocab_size1, output_dim= 100, w
conv_m = Conv1D(298,3, activation = 'relu')(embed_layer)
conv_n = Conv1D(297,3, activation = 'relu')(embed_layer)
conv_o = Conv1D(296,3, activation = 'relu')(embed_layer)

#concatenating layers
layer_out1 = concatenate([conv_m, conv_n, conv_o], axis = -1)
pool_layer1 = MaxPool1D(3)(layer_out1)

conv_i = Conv1D(294,3, activation = 'relu')(pool_layer1)
conv_j = Conv1D(292,3, activation = 'relu')(pool_layer1)
conv_k = Conv1D(290,3, activation = 'relu')(pool_layer1)

#concatenating layers
layer_out2 = concatenate([conv_i, conv_j, conv_k], axis = -1)
pool_layer2 = MaxPool1D(3)(layer_out2)

conv_p = Conv1D(256,3, activation = 'relu')(pool_layer2)

#flatten
flatten = Flatten()(conv_p)
drop_out = Dropout(0.5)(flatten)
dense1 = Dense(60, activation = 'relu', kernel_initializer = HeUnif
output = Dense(20, activation = 'softmax', kernel_initializer = HeU

model = Model([inputs], output)
model.summary()
```

```
Model: "model"

__________________________________________________________________
________________________________
Layer (type)                    Output Shape         Param #     C
onnected to
==================================================================
================================
inputs (InputLayer)             [(None, 300)]        0

__________________________________________________________________
________________________________
embedding (Embedding)           (None, 300, 100)     9862100     i
nputs[0][0]

__________________________________________________________________
________________________________
conv1d (Conv1D)                 (None, 298, 298)     89698       e
mbedding[0][0]

__________________________________________________________________
________________________________
conv1d_1 (Conv1D)               (None, 298, 297)     89397       e
mbedding[0][0]

__________________________________________________________________
________________________________
conv1d_2 (Conv1D)               (None, 298, 296)     89096       e
mbedding[0][0]
```

```
________________________________________________________________________________________________
________________________________
concatenate (Concatenate)       (None, 298, 891)     0           c
onv1d[0][0]
                                                                 c
onv1d_1[0][0]
                                                                 c
onv1d_2[0][0]
________________________________________________________________________________________________
________________________________
max_pooling1d (MaxPooling1D)    (None, 99, 891)      0           c
oncatenate[0][0]
________________________________________________________________________________________________
________________________________
conv1d_3 (Conv1D)               (None, 97, 294)      786156      m
ax_pooling1d[0][0]
________________________________________________________________________________________________
________________________________
conv1d_4 (Conv1D)               (None, 97, 292)      780808      m
ax_pooling1d[0][0]
________________________________________________________________________________________________
________________________________
conv1d_5 (Conv1D)               (None, 97, 290)      775460      m
ax_pooling1d[0][0]
________________________________________________________________________________________________
________________________________
concatenate_1 (Concatenate)     (None, 97, 876)      0           c
onv1d_3[0][0]
                                                                 c
onv1d_4[0][0]

                                                                 c
onv1d_5[0][0]
________________________________________________________________________________________________
________________________________
max_pooling1d_1 (MaxPooling1D)  (None, 32, 876)      0           c
oncatenate_1[0][0]
________________________________________________________________________________________________
________________________________
conv1d_6 (Conv1D)               (None, 30, 256)      673024      m
ax_pooling1d_1[0][0]
________________________________________________________________________________________________
________________________________
flatten (Flatten)               (None, 7680)         0           c
onv1d_6[0][0]
________________________________________________________________________________________________
________________________________
dropout (Dropout)               (None, 7680)         0           f
latten[0][0]
________________________________________________________________________________________________
________________________________
dense (Dense)                   (None, 60)           460860      d
ropout[0][0]
________________________________________________________________________________________________
```

```
________________________________
dense_1 (Dense)                 (None, 20)           1220        d
ense[0][0]
==================================================================
================================
Total params: 13,607,819
Trainable params: 3,745,719
Non-trainable params: 9,862,100
__________________________________________________________________
________________________________
```

In [37]:
```
history_own = LossHistory()
log_dir = "logs/fit/model1_" + datetime.datetime.now().strftime("%Y
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_d

callback_list = [tensorboard_callback, history_own]

model.compile(optimizer = 'adam',loss='categorical_crossentropy', m
```

In [38]:
```
y_pred1 = model.predict(padded_docs_test1)
y_pred1 =(y_pred1>0.5)
```

In [39]:
```
model.fit(padded_docs1, y_train_encoded, batch_size=32, epochs=5, v
```

```
Epoch 1/5
WARNING:tensorflow:Callback method `on_train_batch_begin` is slow
compared to the batch time (batch time: 0.0184s vs `on_train_batch
_begin` time: 0.1249s). Check your callbacks.
WARNING:tensorflow:Callback method `on_train_batch_end` is slow co
mpared to the batch time (batch time: 0.0184s vs `on_train_batch_e
nd` time: 0.0513s). Check your callbacks.
309/309 - 14s - loss: 2.7261 - accuracy: 0.1197 - f1_m: 0.0198 - v
al_loss: 1.9771 - val_accuracy: 0.3111 - val_f1_m: 0.1768
Epoch 2/5
309/309 - 11s - loss: 1.5319 - accuracy: 0.4807 - f1_m: 0.4156 - v
al_loss: 1.1986 - val_accuracy: 0.6084 - val_f1_m: 0.5479
Epoch 3/5
309/309 - 11s - loss: 1.0010 - accuracy: 0.6636 - f1_m: 0.6368 - v
al_loss: 1.0148 - val_accuracy: 0.6816 - val_f1_m: 0.6547
Epoch 4/5
309/309 - 11s - loss: 0.6971 - accuracy: 0.7628 - f1_m: 0.7549 - v
al_loss: 0.9489 - val_accuracy: 0.7014 - val_f1_m: 0.6995
```

Out[39]:
```
<tensorflow.python.keras.callbacks.History at 0x7ff7462ef790>
```

In [40]:
```
loss, accuracy, f1_1 = model.evaluate(padded_docs1, y_train_encoded
```

```
442/442 - 5s - loss: 0.6258 - accuracy: 0.8028 - f1_m: 0.7891
```

In [41]:
```
loss, accuracy, f1_score = model.evaluate(padded_docs_test1, y_test
```

```
148/148 - 2s - loss: 0.9552 - accuracy: 0.7094 - f1_m: 0.7113
```

In [42]:
```
%load_ext tensorboard
```

In [49]:
```
%tensorboard --logdir /content/logs/fit/model1_20210312-161854
```

```
<IPython.core.display.Javascript object>
```

Model 2

In [44]:
```
maxlen = 1200
docs = X_train
t2 = Tokenizer(filters='!"#$%&()*+,-./:;<=>?@[\\]^`{|}~\t\n', num_w
t2.fit_on_texts(docs)
vocab_size2 = len(t2.word_index) + 1
encoded_docs2 = t2.texts_to_sequences(docs)
padded_docs2 = pad_sequences(encoded_docs2, maxlen, padding='post')
```

In [46]:
```
print(len(encoded_docs2[5000]))
```

```
3659
```

In [47]:
```
vocab_size_test2 = len(t2.word_index) + 1
encoded_docs_test2 = t2.texts_to_sequences(X_test)
padded_docs_test2 = pad_sequences(encoded_docs_test2, maxlen, paddi
```

In [50]:
```
embeddings_index = {}
f = open('glove.840B.300d-char.txt')
for line in f:
    values = line.split()
    char = values[0]
    coefs = np.array(values[1:], dtype='float32')
    embeddings_index[char] = coefs
f.close()
print(len(embeddings_index))
```

```
94
```

In [51]:
```
embedding_matrix = np.zeros((vocab_size2, 300))
for word, i in t2.word_index.items():
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector
```

In [68]:
```
model2 = Sequential()
inputs = Input(shape = (maxlen,), name = 'inputs')
embed_layer = Embedding(input_dim = vocab_size2, output_dim= 300, w
conv_m = Conv1D(64,8, activation = 'relu')(embed_layer)
conv_n = Conv1D(32,12, activation = 'relu')(conv_m)

pool_layer1 = MaxPool1D()(conv_n)
```

```
pool_layer1 = MaxPool1D()(conv_h)

conv_i = Conv1D(64,12, activation = 'relu')(pool_layer1)
conv_j = Conv1D(64,12, activation = 'relu')(conv_i)

pool_layer2 = MaxPool1D()(conv_j)

#flatten
flatten = Flatten()(pool_layer2)
drop_out = Dropout(0.5)(flatten)

dense1 = Dense(60, activation = 'relu')(drop_out)
output = Dense(20, activation = 'softmax')(dense1)

model2 = Model([inputs], output)
model2.summary()
```

```
Model: "model_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
inputs (InputLayer)          [(None, 1200)]            0
_________________________________________________________________
embedding_3 (Embedding)      (None, 1200, 300)         22500
_________________________________________________________________
conv1d_15 (Conv1D)           (None, 1193, 64)          153664
_________________________________________________________________
conv1d_16 (Conv1D)           (None, 1182, 32)          24608
_________________________________________________________________
max_pooling1d_6 (MaxPooling1 (None, 591, 32)           0
_________________________________________________________________
conv1d_17 (Conv1D)           (None, 580, 64)           24640
_________________________________________________________________
conv1d_18 (Conv1D)           (None, 569, 64)           49216
_________________________________________________________________
max_pooling1d_7 (MaxPooling1 (None, 284, 64)           0
_________________________________________________________________
flatten_3 (Flatten)          (None, 18176)             0
_________________________________________________________________
dropout_3 (Dropout)          (None, 18176)             0
_________________________________________________________________
dense_6 (Dense)              (None, 60)                1090620
_________________________________________________________________
dense_7 (Dense)              (None, 20)                1220
=================================================================
Total params: 1,366,468
Trainable params: 1,343,968
Non-trainable params: 22,500
_________________________________________________________________
```

In [69]:
```
y_pred2 = model2.predict(padded_docs_test2)
y_pred2 =(y_pred2>0.5)
```

In [70]:
```
history_own = LossHistory()
log_dir = "logs/fit/model2_" + datetime.datetime.now().strftime("%Y
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_d

callback_list = [tensorboard_callback, history_own]

optimizer = tf.keras.optimizers.Adam(learning_rate=0.001, beta_1=0.
model2.compile(optimizer = optimizer,loss='categorical_crossentropy
```

In [71]:
```
model2.fit(padded_docs2, y_train_encoded, batch_size=32, epochs=7,
```

```
Epoch 1/7
WARNING:tensorflow:Callback method `on_train_batch_begin` is slow
compared to the batch time (batch time: 0.0196s vs `on_train_batch
_begin` time: 0.0794s). Check your callbacks.
WARNING:tensorflow:Callback method `on_train_batch_end` is slow co
mpared to the batch time (batch time: 0.0196s vs `on_train_batch_e
nd` time: 0.0589s). Check your callbacks.
309/309 - 11s - loss: 2.9647 - accuracy: 0.0708 - f1_m: 0.0000e+00
- val_loss: 2.9429 - val_accuracy: 0.0762 - val_f1_m: 0.0000e+00
Epoch 2/7
309/309 - 9s - loss: 2.9392 - accuracy: 0.0843 - f1_m: 0.0000e+00
- val_loss: 2.9347 - val_accuracy: 0.0812 - val_f1_m: 0.0000e+00
Epoch 3/7
309/309 - 9s - loss: 2.9311 - accuracy: 0.0867 - f1_m: 0.0000e+00
- val_loss: 2.9362 - val_accuracy: 0.0715 - val_f1_m: 0.0000e+00
Epoch 4/7
309/309 - 10s - loss: 2.9220 - accuracy: 0.0880 - f1_m: 0.0000e+00
- val_loss: 2.9462 - val_accuracy: 0.0807 - val_f1_m: 0.0000e+00
Epoch 5/7
309/309 - 10s - loss: 2.9103 - accuracy: 0.0973 - f1_m: 0.0000e+00
- val_loss: 2.9314 - val_accuracy: 0.0772 - val_f1_m: 0.0000e+00
Epoch 6/7
309/309 - 9s - loss: 2.8948 - accuracy: 0.1023 - f1_m: 9.7491e-04
- val_loss: 2.9366 - val_accuracy: 0.0800 - val_f1_m: 4.5568e-04
Epoch 7/7
309/309 - 9s - loss: 2.8659 - accuracy: 0.1105 - f1_m: 0.0020 - va
l_loss: 2.9290 - val_accuracy: 0.0751 - val_f1_m: 9.1137e-04
```

Out[71]:
```
<tensorflow.python.keras.callbacks.History at 0x7ff75c0d5710>
```

In [72]:
```
loss, accuracy, f1_2 = model2.evaluate(padded_docs2, y_train_encode
```

```
442/442 - 6s - loss: 2.8456 - accuracy: 0.1154 - f1_m: 6.8559e-04
```

In [73]:
```
loss, accuracy, f1_score = model2.evaluate(padded_docs_test2, y_tes
```

```
148/148 - 2s - loss: 2.9309 - accuracy: 0.0826 - f1_m: 0.0012
```

In [74]:
```
%tensorboard --logdir /content/logs/fit/model2_20210312-163646
```

```
<IPython.core.display.Javascript object>
```