RA1911030010030_NitishChaturvedi_FinalAssignment

October 29, 2021

[16]:
```
import time
import numpy as np
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt

import re
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from bs4 import BeautifulSoup
import string

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split

import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras import layers, callbacks
from tensorflow.keras import Model, Sequential
```

[17]:
```
d_train = pd.read_csv("/home/waterupto/Downloads/Corona_NLP_train.csv",
                      encoding='latin1')
d_test = pd.read_csv("/home/waterupto/Downloads/Corona_NLP_test.csv",
                     encoding='latin1')
```

[18]:
```
d_train.head()
```

[18]:
```
   UserName  ScreenName   Location     TweetAt  \
0      3799       48751     London  16-03-2020
1      3800       48752         UK  16-03-2020
2      3801       48753  Vagabonds  16-03-2020
3      3802       48754        NaN  16-03-2020
4      3803       48755        NaN  16-03-2020

                                       OriginalTweet           Sentiment
0  @MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i…             Neutral
```

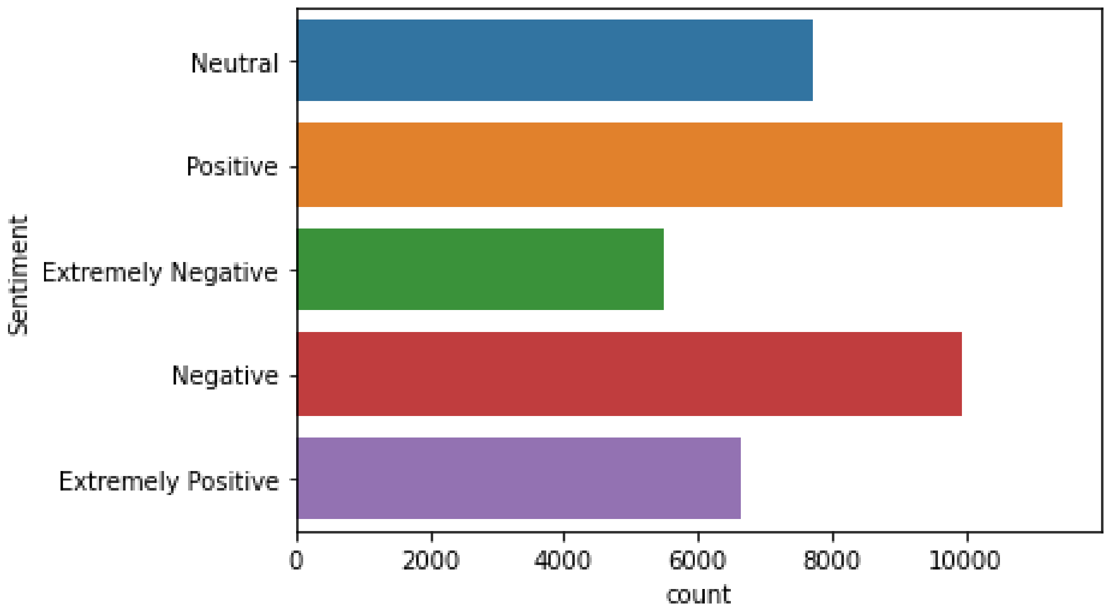
```
1  advice Talk to your neighbours family to excha…           Positive
2  Coronavirus Australia: Woolworths to give elde…           Positive
3  My food stock is not the only one which is emp…           Positive
4  Me, ready to go at supermarket during the #COV…  Extremely Negative
```

```
[19]: d_test.head()
```

```
[19]:    UserName  ScreenName             Location     TweetAt  \
      0         1       44953                  NYC  02-03-2020
      1         2       44954          Seattle, WA  02-03-2020
      2         3       44955                  NaN  02-03-2020
      3         4       44956          Chicagoland  02-03-2020
      4         5       44957  Melbourne, Victoria  03-03-2020

                                             OriginalTweet           Sentiment
      0  TRENDING: New Yorkers encounter empty supermar…  Extremely Negative
      1  When I couldn't find hand sanitizer at Fred Me…            Positive
      2  Find out how you can protect yourself and love…  Extremely Positive
      3  #Panic buying hits #NewYork City as anxious sh…            Negative
      4  #toiletpaper #dunnypaper #coronavirus #coronav…             Neutral
```

```
[20]: sns.countplot(y=d_train.Sentiment)
      plt.show()
```



```
[21]: d_train.isnull().sum()
```

[21]:

```
UserName            0
ScreenName          0
Location         8590
TweetAt             0
OriginalTweet       0
Sentiment           0
dtype: int64
```

[22]:

```
d_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41157 entries, 0 to 41156
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   UserName       41157 non-null  int64 
 1   ScreenName     41157 non-null  int64 
 2   Location       32567 non-null  object
 3   TweetAt        41157 non-null  object
 4   OriginalTweet  41157 non-null  object
 5   Sentiment      41157 non-null  object
dtypes: int64(2), object(4)
memory usage: 1.9+ MB
```

[23]:

```
# Remove unused column
d_train = d_train.drop(['Location','TweetAt','ScreenName'], axis=1)
d_test = d_test.drop(['Location','TweetAt','ScreenName'], axis=1)

d_train.head()
```

[23]:

```
   UserName                                      OriginalTweet  \
0      3799  @MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i…
1      3800  advice Talk to your neighbours family to excha…
2      3801  Coronavirus Australia: Woolworths to give elde…
3      3802  My food stock is not the only one which is emp…
4      3803  Me, ready to go at supermarket during the #COV…

            Sentiment
0             Neutral
1            Positive
2            Positive
3            Positive
4  Extremely Negative
```

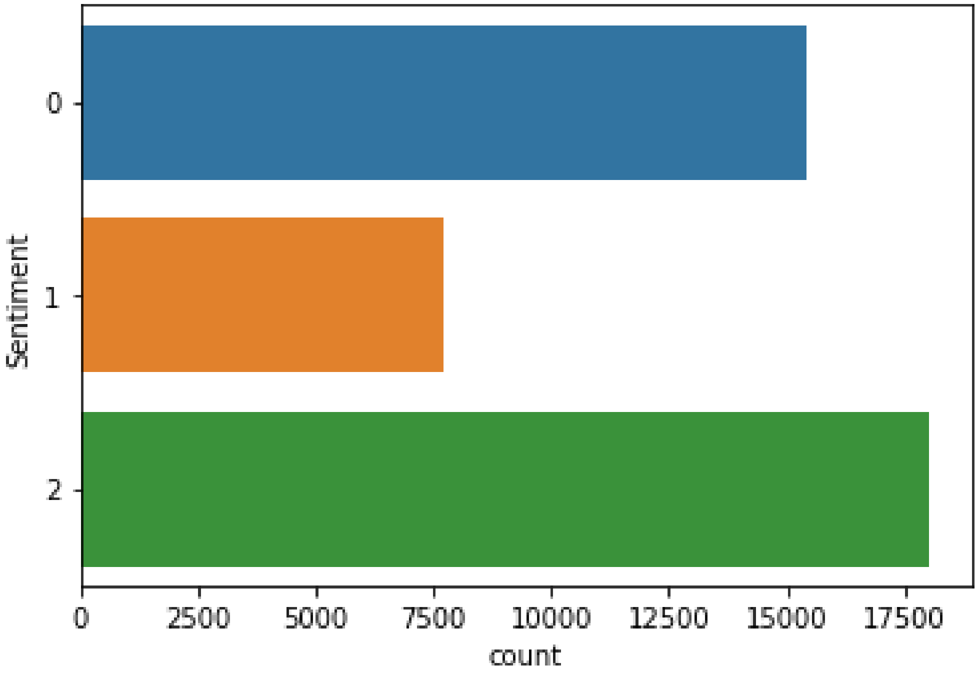
[24]:

```
# Convert sentiment into Positive = 2 , Neutral = 1 , Negative =  0
def convert_Sentiment(label):
    if label == "Extremely Positive":
        return 2
    elif label == "Extremely Negative":
        return 0
    elif label == "Positive":
        return 2
    elif label == "Negative":
        return 0
    else:
        return 1

# Apply convert_Sentiment function
d_train.Sentiment = d_train.Sentiment.apply(lambda x : convert_Sentiment(x))
d_train.head()
```

```
[24]:    UserName                                     OriginalTweet  Sentiment
      0      3799  @MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i…          1
      1      3800  advice Talk to your neighbours family to excha…          2
      2      3801  Coronavirus Australia: Woolworths to give elde…          2
      3      3802  My food stock is not the only one which is emp…          2
      4      3803  Me, ready to go at supermarket during the #COV…          0
```

```
[25]: sns.countplot(y=d_train.Sentiment)
      plt.show()
```



0.0.1 NLP

[26]:

```
def cleaning_text(text):
    stop_words = stopwords.words("english")

    text = re.sub(r'http\S+', " ", text)     # remove urls
    text = re.sub(r'@\w+',' ',text)          # remove mentions
    text = re.sub(r'#\w+', ' ', text)        # remove hastags
    text = re.sub('r<.*?>',' ', text)        # remove html tags

    # remove stopwords
    text = text.split()
    text = " ".join([word for word in text if not word in stop_words])

    for punctuation in string.punctuation:
        text = text.replace(punctuation, "")

    return text

d_train['preprocessing_results'] = d_train['OriginalTweet'].apply(lambda x:␣
 ↪cleaning_text(x))
```

[27]:

```
for i in range(5):
    print('----------------------------------------------')
    random_number=np.random.randint(0,len(d_train)-1)
    print(d_train.preprocessing_results[random_number])
    print('----------------------------------------------\n')
```

```
----------------------------------------------
Another day day 7 toilet paper shop flour eggs Again supermarket like Christmas
Turned round came back home far crowded people got biggest carts lay hands on
People definately still hoarding IMO
----------------------------------------------

----------------------------------------------
Went grocery storeI survive
----------------------------------------------

----------------------------------------------
PSA Wash hands Wash hands Make sure lather soap least 20 seconds If cannot find
water hand sanitizer anti microbial amp 60 alcohol used Not anti bacterial
sanitizer This virus
----------------------------------------------

----------------------------------------------
Shopping people neighborhood canÂ t shouldnÂ t go out As I looking empty shelves
normally stocked I canÂ t help think much food going get thrown away people
donÂ t actually eat it
```

```
----------------------------------------------

----------------------------------------------
With smuggling going on black market cannabis prices absolutely wild right  The
GrowthOp via
----------------------------------------------
```

[28]:

```
# Maximum sentence length
max_len_words = max(list(d_train['preprocessing_results'].apply(len)))
print(max_len_words)
```

```
306
```

[29]:

```
def tokenizer(x_train, y_train, max_len_word):
    # because the data distribution is imbalanced, "stratify" is used
    X_train, X_val, y_train, y_val = train_test_split(x_train, y_train,
                                                      test_size=.2,␣
 ↪shuffle=True,
                                                      stratify=y_train,␣
 ↪random_state=0)

    # Tokenizer
    tokenizer = Tokenizer(num_words=5000)
    tokenizer.fit_on_texts(X_train)
    sequence_dict = tokenizer.word_index
    word_dict = dict((num, val) for (val, num) in sequence_dict.items())

    # Sequence data
    train_sequences = tokenizer.texts_to_sequences(X_train)
    train_padded = pad_sequences(train_sequences,
                                 maxlen=max_len_word,
                                 truncating='post',
                                 padding='post')

    val_sequences = tokenizer.texts_to_sequences(X_val)
    val_padded = pad_sequences(val_sequences,
                                maxlen=max_len_word,
                                truncating='post',
                                padding='post', )

    print(train_padded.shape)
    print(val_padded.shape)
    print('Total words: {}'.format(len(word_dict)))
    return train_padded, val_padded, y_train, y_val, word_dict

X_train, X_val, y_train, y_val, word_dict = tokenizer(d_train.
 ↪preprocessing_results, d_train.Sentiment, 300)
```

```
(32925, 300)
(8232, 300)
Total words: 37419
```

0.0.2 Model

[30]:

```
num_classes = d_train.Sentiment.nunique()
print(num_classes)
```

```
3
```

[31]:

```
model = Sequential([
    layers.Embedding(5000, 300, input_length=300),
    layers.Bidirectional(layers.LSTM(64, return_sequences=True,␣
 ↪recurrent_dropout=0.4)),
    #layers.LSTM(64, return_sequences=True, recurrent_dropout=0.4),
    #layers.BatchNormalization(),
    layers.GlobalAveragePooling1D(),    # or layers.Flatten()
    layers.Dense(64, activation='relu'),
    layers.Dropout(0.4),
    layers.Dense(num_classes, activation='softmax')
])
```

```
2021-10-28 23:49:18.895078: I
tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node
read from SysFS had negative value (-1), but there must be at least one NUMA
node, so returning NUMA node zero
2021-10-28 23:49:18.895533: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
dynamic library 'libcudart.so.11.0'; dlerror: libcudart.so.11.0: cannot open
shared object file: No such file or directory
2021-10-28 23:49:18.895611: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
dynamic library 'libcublas.so.11'; dlerror: libcublas.so.11: cannot open shared
object file: No such file or directory
2021-10-28 23:49:18.895672: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
dynamic library 'libcublasLt.so.11'; dlerror: libcublasLt.so.11: cannot open
shared object file: No such file or directory
2021-10-28 23:49:18.895732: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
dynamic library 'libcufft.so.10'; dlerror: libcufft.so.10: cannot open shared
object file: No such file or directory
2021-10-28 23:49:18.895791: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
dynamic library 'libcurand.so.10'; dlerror: libcurand.so.10: cannot open shared
object file: No such file or directory
2021-10-28 23:49:18.895850: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
```

```
dynamic library 'libcusolver.so.11'; dlerror: libcusolver.so.11: cannot open
shared object file: No such file or directory
2021-10-28 23:49:18.895907: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
dynamic library 'libcusparse.so.11'; dlerror: libcusparse.so.11: cannot open
shared object file: No such file or directory
2021-10-28 23:49:18.895966: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
dynamic library 'libcudnn.so.8'; dlerror: libcudnn.so.8: cannot open shared
object file: No such file or directory
2021-10-28 23:49:18.895986: W
tensorflow/core/common_runtime/gpu/gpu_device.cc:1835] Cannot dlopen some GPU
libraries. Please make sure the missing libraries mentioned above are installed
properly if you would like to use GPU. Follow the guide at
https://www.tensorflow.org/install/gpu for how to download and setup the
required libraries for your platform.
Skipping registering GPU devices…
2021-10-28 23:49:18.896309: I tensorflow/core/platform/cpu_feature_guard.cc:142]
This TensorFlow binary is optimized with oneAPI Deep Neural Network Library
(oneDNN) to use the following CPU instructions in performance-critical
operations:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate
compiler flags.
```

[32]:

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 300, 300)          1500000
_________________________________________________________________
bidirectional (Bidirectional (None, 300, 128)          186880
_________________________________________________________________
global_average_pooling1d (Gl (None, 128)               0
_________________________________________________________________
dense (Dense)                (None, 64)                8256
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 195
=================================================================
Total params: 1,695,331
Trainable params: 1,695,331
Non-trainable params: 0
_________________________________________________________________
```

[33]:

```
model.compile(loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
              metrics=['accuracy'])
```

[34]:

```
start = time.perf_counter()
early_stopping = callbacks.EarlyStopping(monitor ="val_loss",
                                         mode ="min", patience=3)

history = model.fit(X_train, y_train,
                    epochs=50,
                    validation_data=(X_val, y_val),
                    callbacks=[early_stopping],
                    shuffle=True)

elapsed = time.perf_counter() - start
print('Elapsed %.3f seconds.' % elapsed)
```

```
2021-10-28 23:49:27.981935: I
tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] None of the MLIR
Optimization Passes are enabled (registered 2)
```

```
Epoch 1/50
1029/1029 [==============================] - 471s 453ms/step - loss: 0.8351 -
accuracy: 0.6038 - val_loss: 0.4932 - val_accuracy: 0.8254
Epoch 2/50
1029/1029 [==============================] - 730s 710ms/step - loss: 0.4263 -
accuracy: 0.8611 - val_loss: 0.4366 - val_accuracy: 0.8533
Epoch 3/50
1029/1029 [==============================] - 737s 716ms/step - loss: 0.3545 -
accuracy: 0.8866 - val_loss: 0.4531 - val_accuracy: 0.8497
Epoch 4/50
1029/1029 [==============================] - 688s 669ms/step - loss: 0.3042 -
accuracy: 0.9038 - val_loss: 0.4973 - val_accuracy: 0.8446
Epoch 5/50
1029/1029 [==============================] - 578s 562ms/step - loss: 0.2578 -
accuracy: 0.9175 - val_loss: 0.5635 - val_accuracy: 0.8353
Elapsed 3203.753 seconds.
```

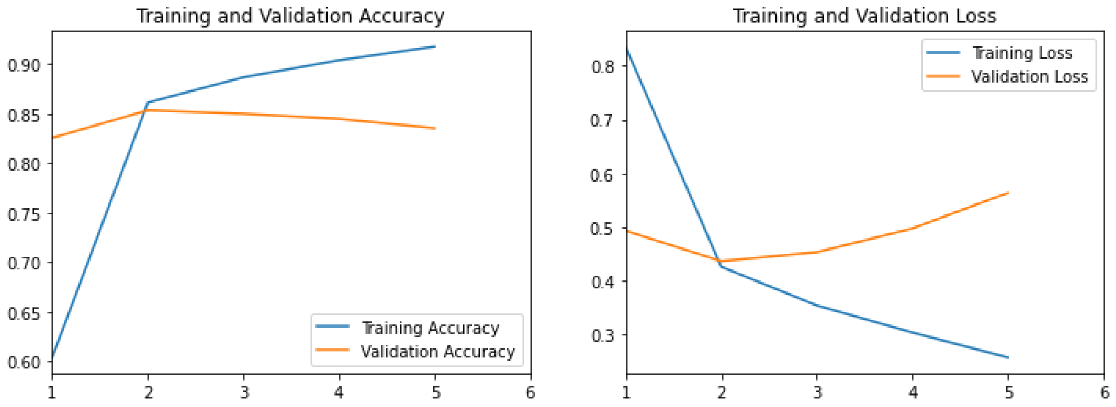
[35]:

```
# Plotting accuracy and val_accuracy
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

loss = history.history['loss']
val_loss = history.history['val_loss']

epochs_range = range(1, len(val_acc)+1)
plt.figure(figsize=(12, 4))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, acc, label='Training Accuracy')
plt.plot(epochs_range, val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.xlim(1, len(val_acc)+1)
plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)
plt.plot(epochs_range, loss, label='Training Loss')
plt.plot(epochs_range, val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.xlim(1, len(val_acc)+1)
plt.title('Training and Validation Loss')
plt.show()
```


Training and Validation Accuracy
Training Accuracy
Validation Accuracy
Training and Validation Loss
Training Loss
Validation Loss