Assignment-4 Team ID-PNT2022TMID42224

LSTM for Text Classification

Importing Required Library

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import keras
import keras.utils
from keras import utils as np_utils
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from keras.models import Model
from keras.layers import LSTM, Activation, Dense, Dropout, Input, Embedding
from keras.optimizers import RMSprop
from keras.preprocessing.text import Tokenizer
from keras.preprocessing import sequence
from keras.utils import to_categorical
from keras.callbacks import EarlyStopping
from keras_preprocessing.sequence import pad_sequences
%matplotlib inline
```

Reading Dataset and Preprocessing

```
df = pd.read_csv('/content/drive/MyDrive/IBM_Assignment_04/spam.csv',delimiter=',',encodin
df.head()
```

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

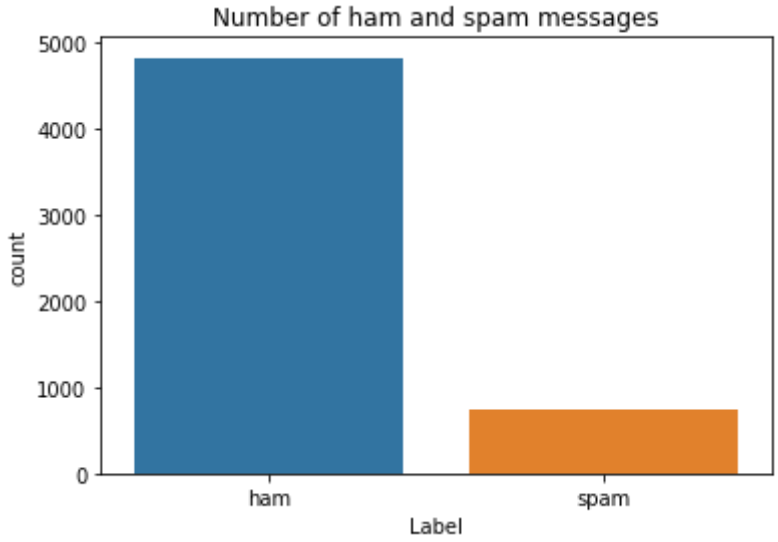
```
df.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1,inplace=True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   v1      5572 non-null   object
 1   v2      5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB
```

```
sns.countplot(df.v1)
plt.xlabel('Label')
plt.title('Number of ham and spam messages')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass
  FutureWarning
Text(0.5, 1.0, 'Number of ham and spam messages')
```



Creating Input and Output Vectors

```
X = df.v2
Y = df.v1
le = LabelEncoder()
Y = le.fit_transform(Y)
Y = Y.reshape(-1,1)
```

Splitting Into Training and Test Data

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.15)
```

Processing The Data

```
from tensorflow.keras.preprocessing.sequence import pad_sequences


max_words = 1000
max_len = 150
tok = Tokenizer(num_words=max_words)
tok.fit_on_texts(X_train)
sequences = tok.texts_to_sequences(X_train)
sequences_matrix = pad_sequences(sequences,maxlen=max_len)
```

Creating The Model and Adding layers

```
def RNN():
 inputs = Input(name='inputs',shape=[max_len])
 layer = Embedding(max_words,50,input_length=max_len)(inputs)
 layer = LSTM(64)(layer)
 layer = Dense(256,name='FC1')(layer)
 layer = Activation('relu')(layer)
 layer = Dropout(0.5)(layer)
 layer = Dense(1,name='out_layer')(layer)
 layer = Activation('sigmoid')(layer)
 model = Model(inputs=inputs,outputs=layer)
 return model
```

Compile The Model

```
model = RNN()
model.summary()
model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 inputs (InputLayer)         [(None, 150)]             0

 embedding (Embedding)       (None, 150, 50)           50000

 lstm (LSTM)                 (None, 64)                29440

 FC1 (Dense)                 (None, 256)               16640

 activation (Activation)     (None, 256)               0

 dropout (Dropout)           (None, 256)               0

 out_layer (Dense)           (None, 1)                 257
```

```
 activation_1 (Activation)   (None, 1)                 0

=================================================================
Total params: 96,337
Trainable params: 96,337
Non-trainable params: 0
_________________________________________________________________
```

Fit The Model

```
model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,
 validation_split=0.2,callbacks=[EarlyStopping(monitor='val_loss',min_delta=0.0001)])
```

```
Epoch 1/10
30/30 [==============================] - 15s 406ms/step - loss: 0.3292 - accuracy: 0
Epoch 2/10
30/30 [==============================] - 8s 265ms/step - loss: 0.0891 - accuracy: 0.9
<keras.callbacks.History at 0x7f98fe902810>
```

Process The Data

```
test_sequences = tok.texts_to_sequences(X_test)
test_sequences_matrix = keras.utils.data_utils.pad_sequences(test_sequences,maxlen=max_len
```

Save The Model

```
model.save('spam.h5')
```

Testing The Model

```
accr = model.evaluate(test_sequences_matrix,Y_test)
```

```
27/27 [==============================] - 1s 27ms/step - loss: 0.0551 - accuracy: 0.98
```

```
print('The Output after Testing the model\n  Loss: {:0.3f}\n  Accuracy: {:0.3f}'.format(ac
```

```
The Output after Testing the model
  Loss: 0.055
  Accuracy: 0.987
```

Colab paid products - Cancel contracts here

✓ 0s completed at 20:41