import pandas as pd import numpy as np import matplotlib.pyplot as plt import seaborn as sns 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 pad\_sequences from keras.utils import to\_categorical from keras.callbacks import EarlyStopping

# READ DATASET AND PRE PROCESSING

df = pd.read\_csv('/content/spam.csv',delimiter=',',encoding='latin-1') df.head()

**Unnamed: Unnamed: Unnamed:**

**v1 v2**

**2 3 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** | h | U dun say so early hor... U c already then | N N | N N | N N |

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

1. = df.v2 Y = df.v1

le = LabelEncoder()

1. = le.fit\_transform(Y)

Y = Y.reshape(-1,1)

X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(X,Y,test\_size=0.15)

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)

# Create Model and Add Layers (LSTM, Dense-(Hidden Layers), Output)

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) model.summary()

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

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Compile the Model

model.compile(loss='binary\_crossentropy',optimizer=RMSprop(),metrics=['accuracy']) Train and Fit the Model

model.fit(sequences\_matrix,Y\_train,batch\_size=128,epochs=10, validation\_split=0.2)

Epoch 1/10

30/30 [==============================] - 12s 286ms/step - loss: 0.3377 - accur

Epoch 2/10

30/30 [==============================] - 9s 301ms/step - loss: 0.0934 - accura

Epoch 3/10

30/30 [==============================] - 10s 327ms/step - loss: 0.0395 - accur

Epoch 4/10

30/30 [==============================] - 9s 317ms/step - loss: 0.0311 - accura

Epoch 5/10

30/30 [==============================] - 9s 294ms/step - loss: 0.0213 - accura

Epoch 6/10

30/30 [==============================] - 9s 305ms/step - loss: 0.0167 - accura

Epoch 7/10

30/30 [==============================] - 9s 316ms/step - loss: 0.0115 - accura

Epoch 8/10

30/30 [==============================] - 9s 286ms/step - loss: 0.0081 - accura

Epoch 9/10

30/30 [==============================] - 9s 310ms/step - loss: 0.0065 - accura

Epoch 10/10

30/30 [==============================] - 10s 346ms/step - loss: 0.0064 - accur

<keras.callbacks.History at 0x7f03f70fe810>



Save The Model

model.save('sms\_classifier.h5')

# Preprocessing the Test Dataset

test\_sequences = tok.texts\_to\_sequences(X\_test)

test\_sequences\_matrix = pad\_sequences(test\_sequences,maxlen=max\_len) Testing the Model

accr = model.evaluate(test\_sequences\_matrix,Y\_test)

27/27 [==============================] - 1s 23ms/step - loss: 0.1346 - accurac

**

0

s

completed at 09:37

[s](https://colab.research.google.com/signup?utm_source=footer&utm_medium=link&utm_campaign=footer_links)

Colab paid product

-

Cancel contracts her

[e](https://colab.research.google.com/cancel-subscription)



print

(

'Test set\n Loss: {:0.3f}\n Accuracy: {:0.3f}'

.

format

(

accr

[

0

[

]

,accr

1

]))

Test set

Loss: 0.135

Accuracy: 0.982