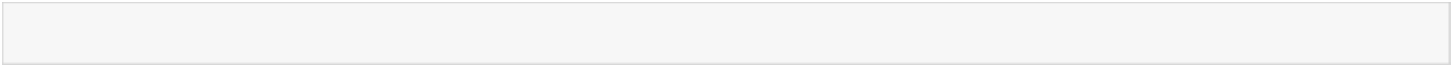
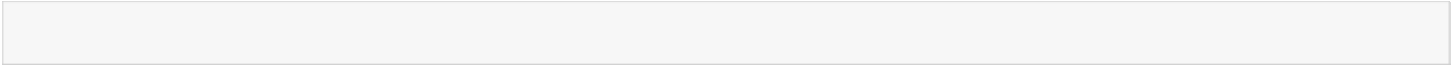
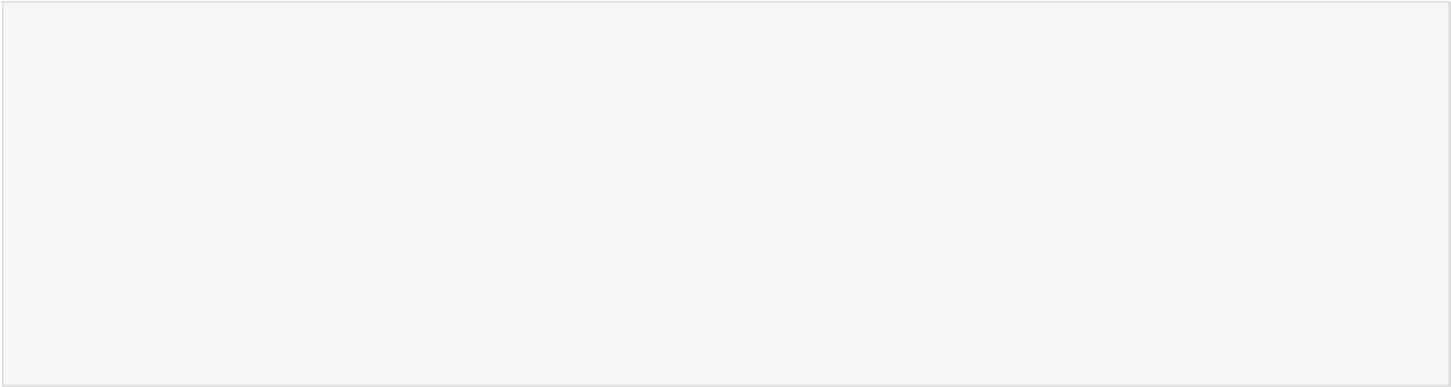
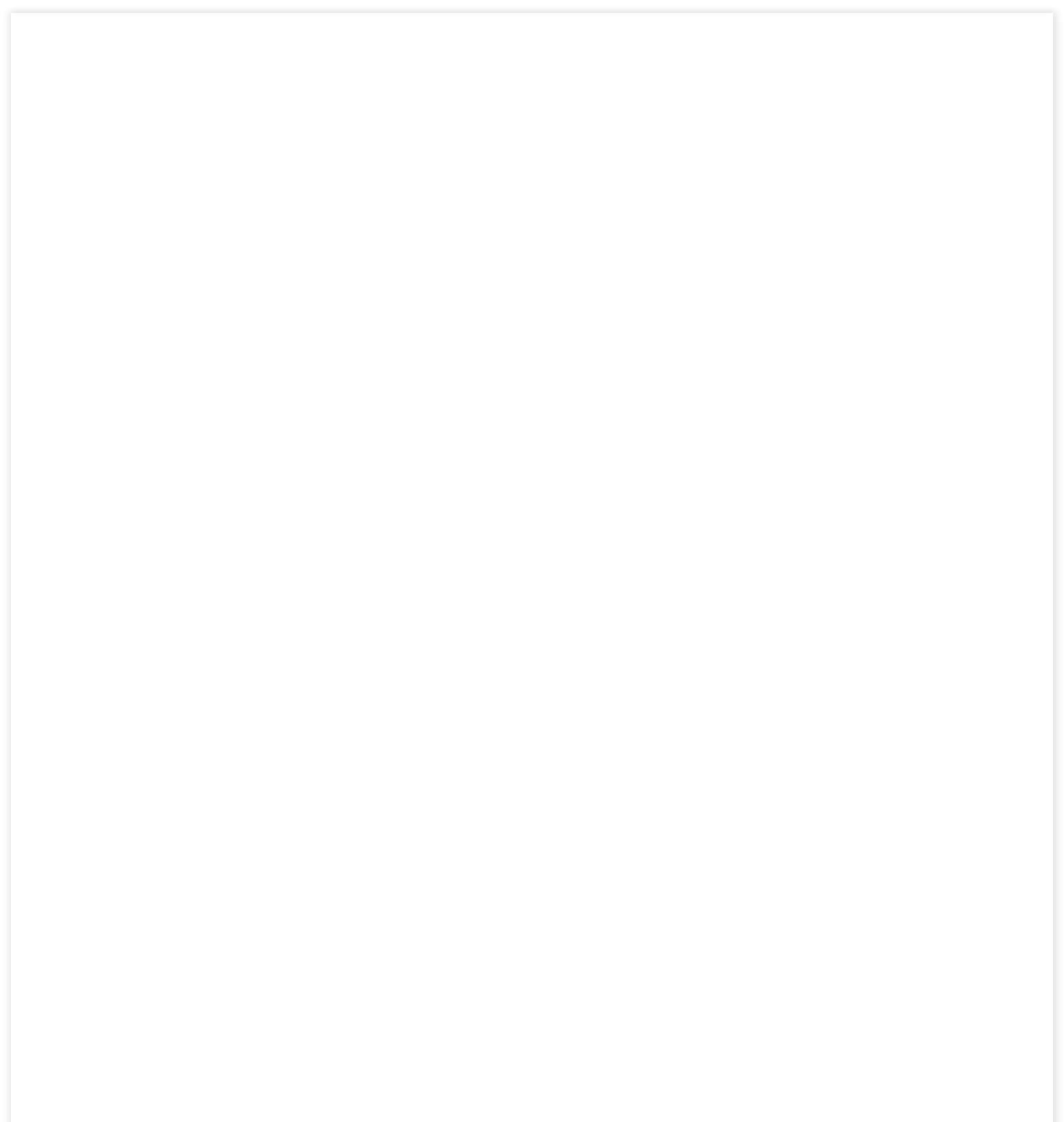
MAHENDRA ENGINEERING COLLEGE FOR WOMEN



NAME:R.VIJAYALAKSHMI CLASS:4-YEAR ECE REG NO:611419106078

SUB:IBM

In [37]:

# #@title Import Libraries

In [38]:

## import pandas as pd import numpy as np import tensorflow as tf

**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** to\_categorical **from keras.utils import** pad\_sequences **from keras.callbacks import** EarlyStopping

%matplotlib inline

In [39]:

# #@title Load the data

In [40]:

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

Out[40]:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **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 |

In [41]:

# #@title Drop unnecessary columns

In [42]:

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 |

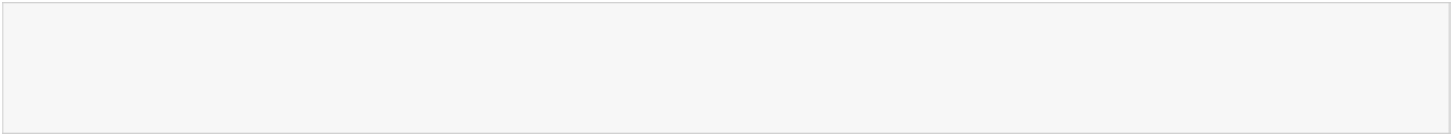
dtypes: object(2) memory usage: 87.2+ KB

In [43]:

# #@title Create input and output vectors and process the labels



In [44]:



X = df.v2 Y = df.v1

le = LabelEncoder()

Y = le.fit\_transform(Y) Y = Y.reshape(-1,1)

In [45]:



*#@title Split the dataset for training and test.*

In [46]:



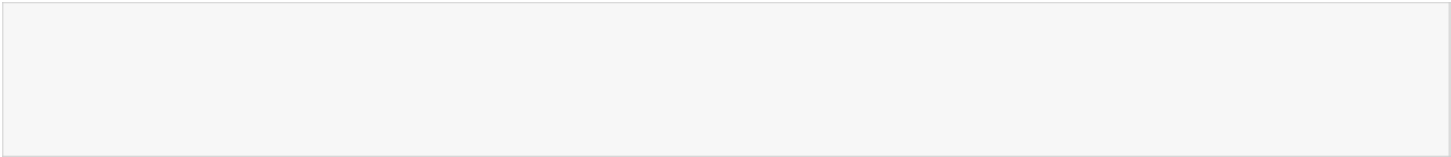
X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(X,Y,test\_size=0.15)

In [47]:



*#@title Process the data*

In [48]:



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)

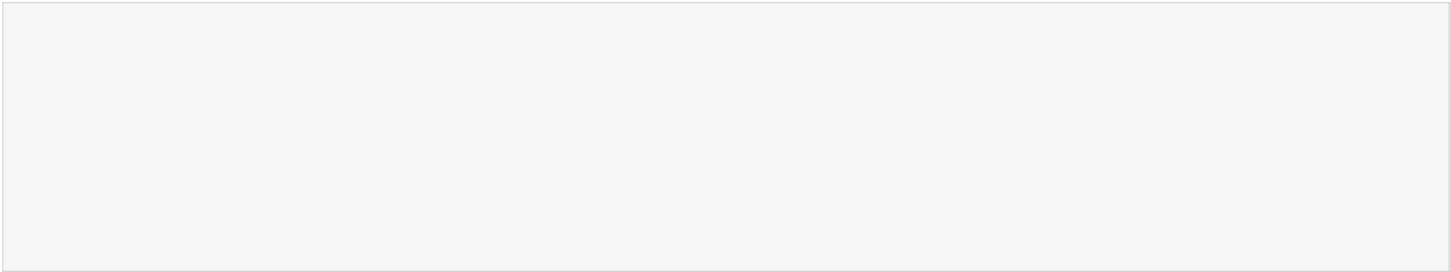
sequences\_matrix =tf.keras.utils.pad\_sequences(sequences,maxlen=max\_len)

In [49]:



*#@title Define the model*

In [50]:



**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)

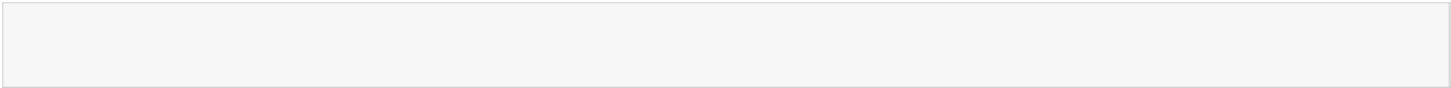
model = Model(inputs=inputs,outputs=layer) **return** model

In [51]:



*#@title Call the function and compile the model*

In [52]:



model = RNN() model.summary()

model.compile(loss='binary\_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])

Model: "model\_1"

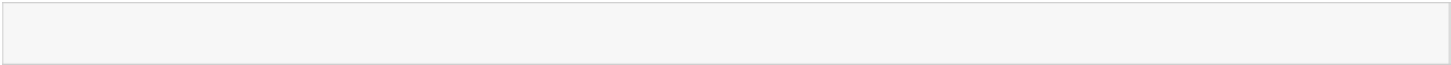
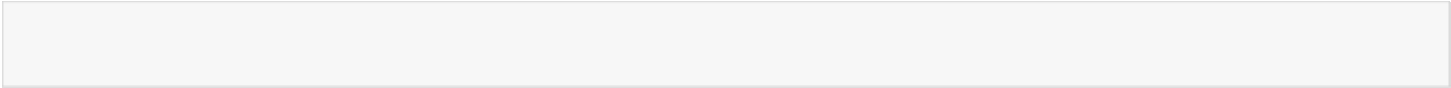
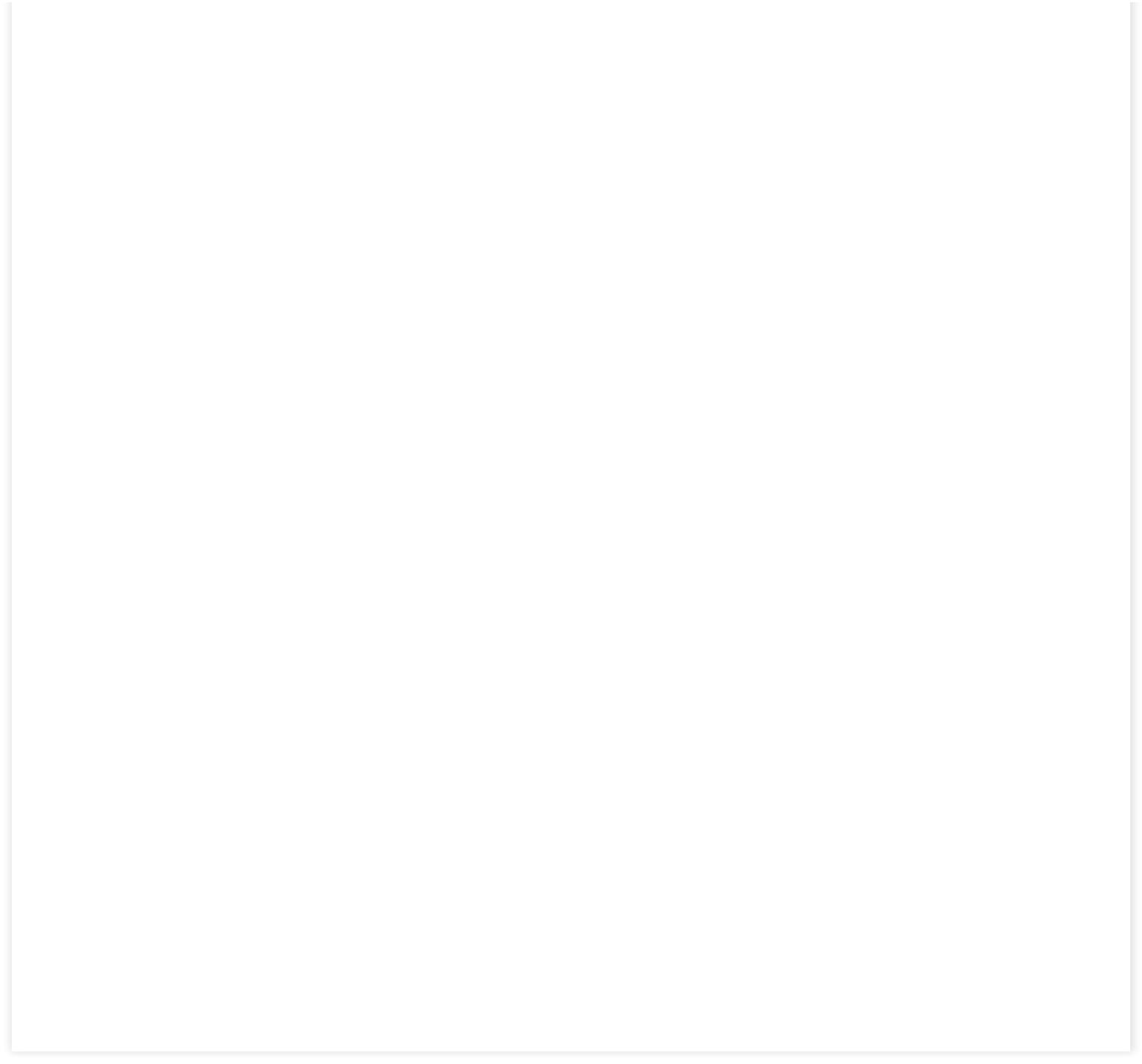
Layer (type) Output Shape Param #

=================================================================

inputs (InputLayer) [(None, 150)] 0

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| embedding\_1 (Embedding) | (None, | 150, | 50) | 50000 |
| lstm\_1 (LSTM) | (None, | 64) |  | 29440 |
| FC1 (Dense) | (None, | 256) |  | 16640 |
| activation\_2 (Activation) | (None, | 256) |  | 0 |

|  |  |  |  |
| --- | --- | --- | --- |
| dropout\_1 (Dropout) | (None, | 256) | 0 |
| out\_layer (Dense) | (None, | 1) | 257 |
| activation\_3 (Activation) | (None, | 1) | 0 |



=================================================================

Total params: 96,337

Trainable params: 96,337

Non-trainable params: 0

In [53]:

# #@title Fit the model

In [54]:

model.fit(sequences\_matrix,Y\_train,batch\_size=128,epochs=10, validation\_split=0.2,callbacks=[EarlyStopping(monitor='val\_loss',min\_delta=0.0

001)])

Epoch 1/10

30/30 [==============================] - 10s 267ms/step - loss: 0.3345 - accuracy: 0.8730

* val\_loss: 0.1491 - val\_accuracy: 0.9462 Epoch 2/10

30/30 [==============================] - 8s 251ms/step - loss: 0.0887 - accuracy: 0.9794

* val\_loss: 0.0625 - val\_accuracy: 0.9821 Out[54]:

<keras.callbacks.History at 0x7f0a5c167750>

In [55]:

# #@title Process the test data

In [56]:

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

test\_sequences\_matrix = tf.keras.utils.pad\_sequences(test\_sequences,maxlen=max\_len)

In [57]:

# #@title Evaluate the model with the test

In [58]:

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

27/27 [==============================] - 1s 21ms/step - loss: 0.0643 - accuracy: 0.9797

In [59]:

**print**('Test set**\n** Loss: {:0.3f}**\n** Accuracy: {:0.3f}'.format(accr[0],accr[1])) Test set

Loss: 0.064

Accuracy: 0.980