

IMPORT THE LIBRARIES

```
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 DO PREPROCESSING

READ DATASET

```
ag = pd.read_csv('/content/spam.csv', delimiter=',', encoding='latin-1')
ag.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

PREPROCESSING THE DATASET

```
ag.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype
---  -
0   v1               5572 non-null  object
1   v2               5572 non-null  object
2   Unnamed: 2       50 non-null    object
3   Unnamed: 3       12 non-null    object
4   Unnamed: 4       6 non-null     object
```

```
dtypes: object(5)
memory usage: 217.8+ KB
```

```
X = ag.v2
Y = ag.v1
le = LabelEncoder()
Y = 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)
```

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 279ms/step - loss: 0.3311 - accuracy: 0
Epoch 2/10
30/30 [=====] - 8s 260ms/step - loss: 0.0839 - accuracy: 0.9
Epoch 3/10
30/30 [=====] - 12s 417ms/step - loss: 0.0446 - accuracy: 0
Epoch 4/10
30/30 [=====] - 10s 318ms/step - loss: 0.0348 - accuracy: 0
Epoch 5/10
30/30 [=====] - 9s 304ms/step - loss: 0.0275 - accuracy: 0.9
Epoch 6/10
30/30 [=====] - 8s 259ms/step - loss: 0.0228 - accuracy: 0.9
Epoch 7/10
30/30 [=====] - 10s 348ms/step - loss: 0.0176 - accuracy: 0
Epoch 8/10
30/30 [=====] - 9s 293ms/step - loss: 0.0147 - accuracy: 0.9
Epoch 9/10
30/30 [=====] - 10s 337ms/step - loss: 0.0123 - accuracy: 0
Epoch 10/10
30/30 [=====] - 12s 380ms/step - loss: 0.0098 - accuracy: 0
<keras.callbacks.History at 0x7f8788200850>

```

SAVE THE MODEL

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

PREPROCESSING THE 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 22ms/step - loss: 0.0715 - accuracy: 0.98



```
print('Test set\n Loss: {:.3f}\n Accuracy: {:.3f}'.format(accr[0],accr[1]))
```

```
Test set  
Loss: 0.072  
Accuracy: 0.983
```

[Colab paid products](#) - [Cancel contracts here](#)

