```
import pandas as pd
import numpy
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential, load_model
from keras.layers import Dense, Embedding, LSTM, SpatialDropout1D
from sklearn.model_selection import train_test_split
from keras.utils.np_utils import to_categorical
import re
from sklearn.preprocessing import LabelEncoder
from keras.callbacks import TensorBoard

data = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Sentiment.csv')
data = data[['text', 'sentiment']]

data['text'] = data['text'].apply(lambda x: x.lower())
data['text'] = data['text'].apply((lambda x: re.sub('[^a-zA-z0-9\s]', '', x)))

print(data[data['sentiment'] == 'Positive'].size)
print(data[data['sentiment'] == 'Negative'].size)
print(data[data['sentiment'] == 'Neutral'].size)

for idx, row in data.iterrows():
    row[0] = row[0].replace('rt', ' ')

max_features = 2000
tokenizer = Tokenizer(num_words=max_features, split=' ')
tokenizer.fit_on_texts(data['text'].values)
X = tokenizer.texts_to_sequences(data['text'].values)
X = pad_sequences(X, maxlen=28)

embed_dim = 128
lstm_out = 196

labelencoder = LabelEncoder()
integer_encoded = labelencoder.fit_transform(data['sentiment'])
y = to_categorical(integer_encoded)
X_train, X_test, Y_train, Y_test = train_test_split(X, y, test_size=0.25, random_state=42)
print(X_train.shape, Y_train.shape)
print(X_test.shape, Y_test.shape)

batch_size = 128
model = Sequential()
model.add(Embedding(max_features, embed_dim, input_length=X.shape[1]))
model.add(SpatialDropout1D(0.4))
model.add(LSTM(lstm_out, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(3, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
tb = TensorBoard(log_dir="logs/{}", histogram_freq=0, write_graph=True, write_images=True)
model.fit(X_train, Y_train, epochs=5, batch_size=batch_size, verbose=2, callbacks=[tb])

model.save('/content/drive/MyDrive/Colab Notebooks/model.h5')
m = load_model('/content/drive/MyDrive/Colab Notebooks/model.h5')

print(m.summary())
```

```
text = [['A lot of good things are happening. We are respected again throughout the world, and
         'thing.@realDonaldTrump']]
df = pd.DataFrame(text, index=range(0, 1, 1), columns=list('t'))
df['t'] = df['t'].apply(lambda x: x.lower())
df['t'] = df['t'].apply((lambda x: re.sub('[^a-zA-z0-9\s]', '', x)))
max_fatures = 2000
tokenizer = Tokenizer(num_words=max_fatures, split=' ')
tokenizer.fit_on_texts(df['t'].values)
X = tokenizer.texts_to_sequences(df['t'].values)
X = pad_sequences(X, maxlen=28)

output = m.predict(X)
print(output)
print(numpy.where(max(output[0])), ":", (max(output[0])))
print(numpy.argmax(output))
print(model.summary())
```

```
4472
16986
6284
tokenizer.texts_to_sequences [[52, 78, 341, 456, 22, 2, 420, 365, 95, 29, 51, 1039, 1],

 pad_sequences
 [[   0    0    0 ...   51 1039    1]
 [   0    0    0 ... 1577 1356  847]
 [   0    0    0 ...   10  696  518]
 ...
 [   0    0    0 ...   68   62    3]
 [   0    0    0 ... 1112 1588   81]
 [   0    0    0 ...  196    3  880]]
(10403, 28) (10403, 3)
(3468, 28) (3468, 3)
Epoch 1/5
82/82 - 31s - loss: 0.8819 - accuracy: 0.6189
Epoch 2/5
82/82 - 28s - loss: 0.7368 - accuracy: 0.6843
Epoch 3/5
82/82 - 29s - loss: 0.6644 - accuracy: 0.7245
Epoch 4/5
82/82 - 29s - loss: 0.6259 - accuracy: 0.7310
Epoch 5/5
82/82 - 30s - loss: 0.5962 - accuracy: 0.7445
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 28, 128)           256000
_________________________________________________________________
spatial_dropout1d (SpatialDr (None, 28, 128)           0
_________________________________________________________________
lstm_1 (LSTM)                (None, 196)               254800
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 591
=================================================================
Total params: 511,391
Trainable params: 511,391
Non-trainable params: 0
_________________________________________________________________
None
[[0.58673155 0.09857924 0.31468922]]
(array([0]),) : 0.58673155
0
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 28, 128)           256000
_________________________________________________________________
spatial_dropout1d (SpatialDr (None, 28, 128)           0
_________________________________________________________________
lstm_1 (LSTM)                (None, 196)               254800
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 591
=================================================================
Total params: 511,391
Trainable params: 511,391
```