

# Neural Networks & Deep Learning

## Assignment -6 (ICP\_6)

Name : Sravani Mannuru

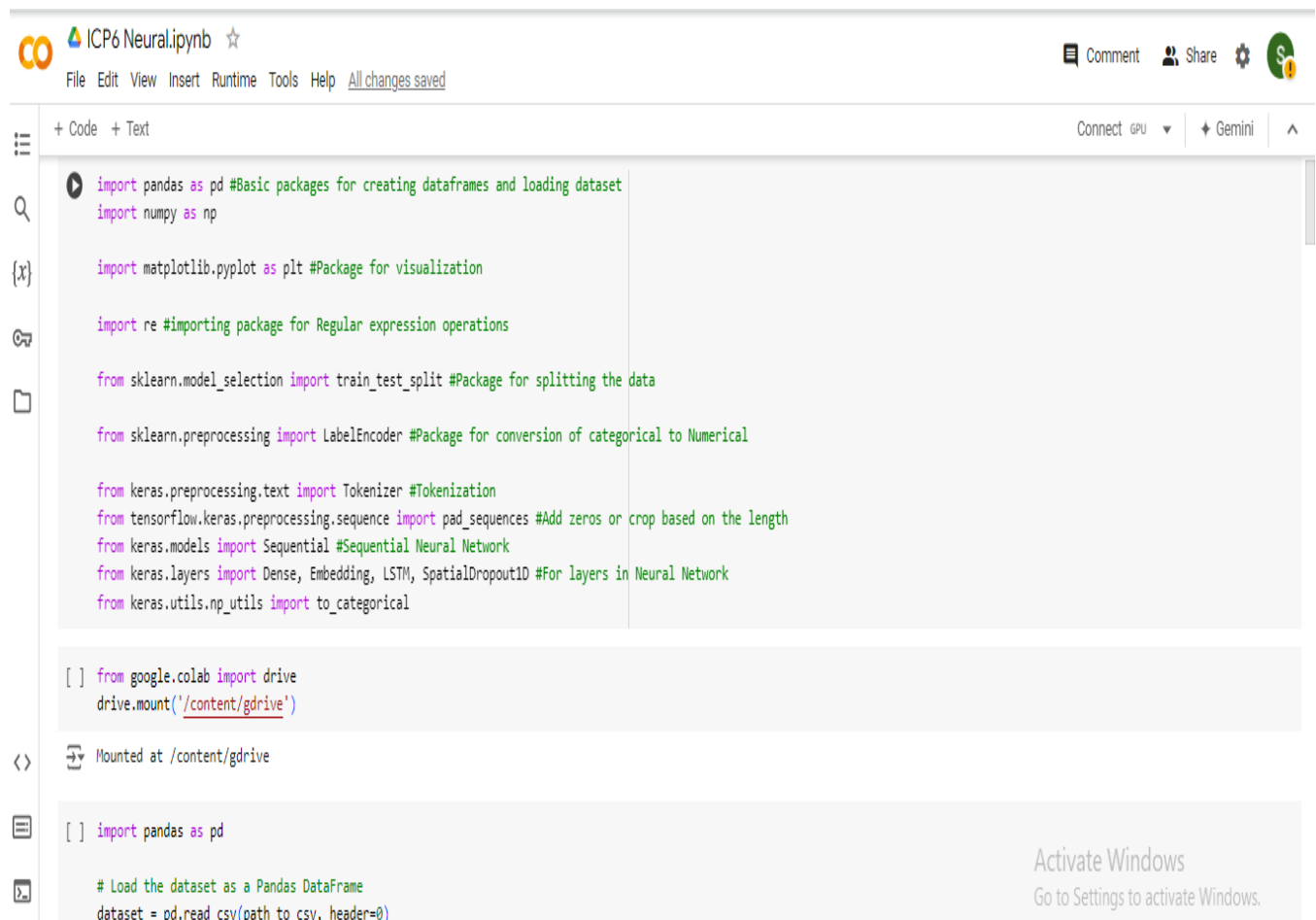
Student Id : 700766162

Github link : <https://github.com/sravs2031/Neural-Networks-ICP6.git>

Video link:

[https://drive.google.com/file/d/1z0mh39wsYGCdOG92DNjGKDNd0EUz5aYd/view?usp=drive\\_link](https://drive.google.com/file/d/1z0mh39wsYGCdOG92DNjGKDNd0EUz5aYd/view?usp=drive_link)

Screenshots :



The screenshot displays a Jupyter Notebook titled "ICP6 Neural.ipynb". The interface includes a top menu bar with options like File, Edit, View, Insert, Runtime, Tools, and Help. Below the menu, there's a toolbar with icons for code and text input, and a status bar indicating "All changes saved". The notebook content is divided into two cells. The first cell contains a series of import statements for various libraries: pandas, numpy, matplotlib, re, sklearn, keras, and tensorflow. The second cell contains code to mount a Google Drive folder and import pandas. A watermark "Activate Windows" is visible in the bottom right corner.

```
import pandas as pd #Basic packages for creating dataframes and loading dataset
import numpy as np

import matplotlib.pyplot as plt #Package for visualization

import re #importing package for Regular expression operations

from sklearn.model_selection import train_test_split #Package for splitting the data

from sklearn.preprocessing import LabelEncoder #Package for conversion of categorical to Numerical

from keras.preprocessing.text import Tokenizer #Tokenization
from tensorflow.keras.preprocessing.sequence import pad_sequences #Add zeros or crop based on the length
from keras.models import Sequential #Sequential Neural Network
from keras.layers import Dense, Embedding, LSTM, SpatialDropout1D #For layers in Neural Network
from keras.utils.np_utils import to_categorical

[ ] from google.colab import drive
drive.mount('/content/gdrive')

[ ] import pandas as pd

# Load the dataset as a Pandas DataFrame
dataset = pd.read_csv(path_to_csv, header=0)
```

ICP6 Neural.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Gemini

+ Code + Text Connect GPU Gemini

```
[ ] import pandas as pd

# Load the dataset as a Pandas DataFrame
dataset = pd.read_csv(path_to_csv, header=0)

# Select only the necessary columns 'text' and 'sentiment'
mask = dataset.columns.isin(['text', 'sentiment'])
data = dataset.loc[:, mask]

# Keeping only the necessary columns

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

<ipython-input-29-cee1da567eb8>:1: SettingWithCopyWarning:  
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)  
data['text'] = data['text'].apply(lambda x: x.lower())

<ipython-input-29-cee1da567eb8>:2: SettingWithCopyWarning:  
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)  
data['text'] = data['text'].apply(lambda x: re.sub('[^a-zA-z0-9\s]', '', x)))

Activate Windows  
Go to Settings to activate Windows.

ICP6 Neural.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Gemini

+ Code + Text Connect GPU Gemini

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
data['text'] = data['text'].apply(lambda x: re.sub('[^a-zA-z0-9\s]', '', x)))

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

[ ] max_fatures = 2000
tokenizer = Tokenizer(num_words=max_fatures, split=' ') #Maximum words is 2000 to tokenize sentence
tokenizer.fit_on_texts(data['text'].values)
X = tokenizer.texts_to_sequences(data['text'].values) #taking values to feature matrix

[ ] X = pad_sequences(X) #Padding the feature matrix

embed_dim = 128 #Dimension of the Embedded layer
lstm_out = 196 #Long short-term memory (LSTM) layer neurons

[ ] def createmodel():
    model = Sequential() #Sequential Neural Network
    model.add(Embedding(max_fatures, embed_dim,input_length = X.shape[1])) #input dimension 2000 Neurons, output dimension 128 Neurons
    model.add(LSTM(lstm_out, dropout=0.2, recurrent_dropout=0.2)) #Drop out 20%, 196 output Neurons, recurrent dropout 20%
    model.add(Dense(3,activation='softmax')) #3 output neurons[positive, Neutral, Negative], softmax as activation
    model.compile(loss = 'categorical_crossentropy', optimizer='adam',metrics = ['accuracy']) #Compiling the model
    return model
```

Activate Windows  
Go to Settings to activate Windows.

ICP6 Neural.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Gemini

+ Code + Text Connect GPU Gemini

```
[ ] def createmodel():
    model = Sequential() #Sequential Neural Network
    model.add(Embedding(max_fatures, embed_dim,input_length = X.shape[1])) #input dimension 2000 Neurons, output dimension 128 Neurons
    model.add(LSTM(lstm_out, dropout=0.2, recurrent_dropout=0.2)) #Drop out 20%, 196 output Neurons, recurrent dropout 20%
    model.add(Dense(3,activation='softmax')) #3 output neurons[positive, Neutral, Negative], softmax as activation
    model.compile(loss = 'categorical_crossentropy', optimizer='adam',metrics = ['accuracy']) #Compiling the model
    return model
# print(model.summary())

[ ] labelencoder = LabelEncoder() #Applying Label Encoding on the label matrix
integer_encoded = labelencoder.fit_transform(data['sentiment']) #fitting the model
y = to_categorical(integer_encoded)
X_train, X_test, Y_train, Y_test = train_test_split(X,y, test_size = 0.33, random_state = 42) #67% training data, 33% test data split

[ ] batch_size = 32 #Batch size 32
model = createmodel() #Function call to Sequential Neural Network
model.fit(X_train, Y_train, epochs = 1, batch_size=batch_size, verbose = 2) #verbose the higher, the more messages
score,acc = model.evaluate(X_test,Y_test,verbose=2,batch_size=batch_size) #evaluating the model
print(score)
print(acc)
```

WARNING:tensorflow:Layer lstm will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
291/291 - 56s - loss: 0.8208 - accuracy: 0.6530 - 56s/epoch - 193ms/step  
144/144 - 2s - loss: 0.7517 - accuracy: 0.6796 - 2s/epoch - 11ms/step  
0.751739501953125  
0.6795544028282166

Activate Windows  
Go to Settings to activate Windows.

ICP6 Neural.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Gemini

+ Code + Text Connect GPU Gemini

```
0.751739501953125
0.6795544028282166

[ ] print(model.metrics_names) #metrics of the model
['loss', 'accuracy']

1. Save the model and use the saved model to predict on new text data (ex, "A lot of
good things are happening. We are respected again throughout the world, and that's a
great thing.@realDonaldTrump")


[ ] model.save('sentimentAnalysis.h5') #Saving the model

[ ] from keras.models import load_model #Importing the package for importing the saved model
model= load_model('sentimentAnalysis.h5') #loading the saved model
```

WARNING:tensorflow:Layer lstm will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.

```
[ ] print(integer_encoded)
print(data['sentiment'])
```

Activate Windows  
Go to Settings to activate Windows.

 ICP6 Neural.ipynb ☆

File Edit View Insert Runtime Tools Help [All changes saved](#)

+ Code + Text

Connect GPU Gemini ^

1. Save the model and use the saved model to predict on new text data (ex, "A lot of good things are happening. We are respected again throughout the world, and that's a great thing.@realDonaldTrump")

```
[ ] model.save('sentimentAnalysis.h5') #Saving the model
```


```
from keras.models import load_model #Importing the package for importing the saved model
model= load_model('sentimentAnalysis.h5') #loading the saved model
```

WARNING:tensorflow:Layer lstm will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.

```
[ ] print(integer_encoded)
print(data['sentiment'])
```

```
[ ] [1 2 1 ... 2 0 2]
0 Neutral
1 Positive
2 Neutral
3 Positive
4 Positive
...
13866 Negative
```

Activate Windows  
Go to Settings to activate Windows.

 ICP6 Neural.ipynb ☆

File Edit View Insert Runtime Tools Help [All changes saved](#)

+ Code + Text

Connect GPU Gemini ^

```
[ ] 13866 Negative
13867 Positive
13868 Positive
13869 Negative
13870 Positive
Name: sentiment, Length: 13871, dtype: object
```

```
[ ] # Predicting on the text data
sentence = ['A lot of good things are happening. We are respected again throughout the world, and that is a great thing.@realDonaldTrump']
sentence = tokenizer.texts_to_sequences(sentence) # Tokenizing the sentence
sentence = pad_sequences(sentence, maxlen=28, dtype='int32', value=0) # Padding the sentence
sentiment_probs = model.predict(sentence, batch_size=1, verbose=2)[0] # Predicting the sentence text
sentiment = np.argmax(sentiment_probs)

print(sentiment_probs)
if sentiment == 0:
    print("Neutral")
elif sentiment < 0:
    print("Negative")
elif sentiment > 0:
    print("Positive")
else:
    print("Cannot be determined")
```

1/1 - 0s - 22ms/epoch - 22ms/step  
[0.3347626 0.16386913 0.5013683 ]

Activate Windows  
Go to Settings to activate Windows.

ICP6 Neural.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Gemini

+ Code + Text

Connect GPU Gemini

[ ] 1/1 - 0s - 22ms/epoch - 22ms/step  
[0.3347626 0.16386913 0.5013683 ]  
Positive

2. Apply GridSearchCV on the source code provided in the class

```
from keras.wrappers.scikit_learn import KerasClassifier #importing Keras classifier
from sklearn.model_selection import GridSearchCV #importing Grid search CV

model = KerasClassifier(build_fn=create_model, verbose=2) #initiating model to test performance by applying multiple hyper parameters
batch_size = [10, 20, 40] #hyper parameter batch_size
epochs = [1, 2] #hyper parameter no. of epochs
param_grid = {'batch_size': batch_size, 'epochs': epochs} #creating dictionary for batch size, no. of epochs
grid = GridSearchCV(estimator=model, param_grid=param_grid) #Applying dictionary with hyper parameters
grid_result = grid.fit(X_train, Y_train) #Fitting the model
# summarize results
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_)) #best score, best hyper parameters
```

<ipython-input-45-6c99b49150f4>:4: DeprecationWarning: KerasClassifier is deprecated, use Sci-Keras (https://github.com/adriangb/scikeras) instead. See https://www.adriangb.com/scikeras/
model = KerasClassifier(build\_fn=create\_model, verbose=2) #initiating model to test performance by applying multiple hyper parameters
WARNING:tensorflow:Layer lstm\_1 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.
744/744 - 108s - loss: 0.8243 - accuracy: 0.6433 - 108s/epoch - 145ms/step
186/186 - 2s - loss: 0.7794 - accuracy: 0.6681 - 2s/epoch - 12ms/step
WARNING:tensorflow:Layer lstm\_2 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.
744/744 - 106s - loss: 0.8200 - accuracy: 0.6476 - 106s/epoch - 143ms/step

Activate Windows  
Go to Settings to activate Windows.

ICP6 Neural.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Gemini

+ Code + Text

Connect GPU Gemini

744/744 - 107s - loss: 0.8203 - accuracy: 0.6440 - 107s/epoch - 143ms/step  
186/186 - 2s - loss: 0.7734 - accuracy: 0.6679 - 2s/epoch - 11ms/step  
WARNING:tensorflow:Layer lstm\_6 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
Epoch 1/2  
744/744 - 108s - loss: 0.8251 - accuracy: 0.6481 - 108s/epoch - 145ms/step  
Epoch 2/2  
744/744 - 96s - loss: 0.6777 - accuracy: 0.7098 - 96s/epoch - 129ms/step  
186/186 - 2s - loss: 0.7344 - accuracy: 0.6902 - 2s/epoch - 12ms/step  
WARNING:tensorflow:Layer lstm\_7 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
Epoch 1/2  
744/744 - 105s - loss: 0.8208 - accuracy: 0.6488 - 105s/epoch - 141ms/step  
Epoch 2/2  
744/744 - 95s - loss: 0.6808 - accuracy: 0.7127 - 95s/epoch - 127ms/step  
186/186 - 3s - loss: 0.7464 - accuracy: 0.6778 - 3s/epoch - 16ms/step  
WARNING:tensorflow:Layer lstm\_8 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
Epoch 1/2  
744/744 - 108s - loss: 0.8200 - accuracy: 0.6455 - 108s/epoch - 145ms/step  
Epoch 2/2  
744/744 - 96s - loss: 0.6682 - accuracy: 0.7186 - 96s/epoch - 130ms/step  
186/186 - 2s - loss: 0.7458 - accuracy: 0.6864 - 2s/epoch - 11ms/step  
WARNING:tensorflow:Layer lstm\_9 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
Epoch 1/2  
744/744 - 107s - loss: 0.8252 - accuracy: 0.6452 - 107s/epoch - 144ms/step  
Epoch 2/2  
744/744 - 95s - loss: 0.6764 - accuracy: 0.7123 - 95s/epoch - 128ms/step  
186/186 - 2s - loss: 0.7443 - accuracy: 0.6712 - 2s/epoch - 11ms/step  
WARNING:tensorflow:Layer lstm\_10 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
Epoch 1/2  
744/744 - 105s - loss: 0.8182 - accuracy: 0.6490 - 105s/epoch - 141ms/step  
Epoch 2/2

Activate Windows  
Go to Settings to activate Windows.

ICP6 Neural.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Gemini

+ Code + Text Connect GPU Gemini

Epoch 1/2  
744/744 - 105s - loss: 0.8182 - accuracy: 0.6490 - 105s/epoch - 141ms/step  
Epoch 2/2  
744/744 - 94s - loss: 0.6692 - accuracy: 0.7143 - 94s/epoch - 127ms/step  
186/186 - 2s - loss: 0.7689 - accuracy: 0.6749 - 2s/epoch - 11ms/step  
WARNING:tensorflow:Layer lstm\_11 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
372/372 - 61s - loss: 0.8300 - accuracy: 0.6429 - 61s/epoch - 165ms/step  
93/93 - 1s - loss: 0.7640 - accuracy: 0.6606 - 1s/epoch - 12ms/step  
WARNING:tensorflow:Layer lstm\_12 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
372/372 - 59s - loss: 0.8303 - accuracy: 0.6438 - 59s/epoch - 160ms/step  
93/93 - 1s - loss: 0.7571 - accuracy: 0.6794 - 1s/epoch - 14ms/step  
WARNING:tensorflow:Layer lstm\_13 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
372/372 - 59s - loss: 0.8337 - accuracy: 0.6450 - 59s/epoch - 158ms/step  
93/93 - 1s - loss: 0.7684 - accuracy: 0.6735 - 1s/epoch - 12ms/step  
WARNING:tensorflow:Layer lstm\_14 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
372/372 - 58s - loss: 0.8267 - accuracy: 0.6398 - 58s/epoch - 157ms/step  
93/93 - 2s - loss: 0.7480 - accuracy: 0.6787 - 2s/epoch - 18ms/step  
WARNING:tensorflow:Layer lstm\_15 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
372/372 - 58s - loss: 0.8273 - accuracy: 0.6482 - 58s/epoch - 155ms/step  
93/93 - 2s - loss: 0.7958 - accuracy: 0.6642 - 2s/epoch - 18ms/step  
WARNING:tensorflow:Layer lstm\_16 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
Epoch 1/2  
372/372 - 59s - loss: 0.8283 - accuracy: 0.6447 - 59s/epoch - 159ms/step  
Epoch 2/2  
372/372 - 48s - loss: 0.6820 - accuracy: 0.7147 - 48s/epoch - 129ms/step  
93/93 - 1s - loss: 0.7243 - accuracy: 0.6907 - 1s/epoch - 12ms/step  
WARNING:tensorflow:Layer lstm\_17 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
Epoch 1/2  
372/372 - 59s - loss: 0.8281 - accuracy: 0.6407 - 59s/epoch - 158ms/step

ICP6 Neural.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Gemini

+ Code + Text Connect GPU Gemini

Epoch 2/2  
186/186 - 25s - loss: 0.6936 - accuracy: 0.7010 - 25s/epoch - 136ms/step  
47/47 - 1s - loss: 0.7462 - accuracy: 0.6837 - 730ms/epoch - 16ms/step  
WARNING:tensorflow:Layer lstm\_28 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
Epoch 1/2  
186/186 - 38s - loss: 0.8465 - accuracy: 0.6363 - 38s/epoch - 202ms/step  
Epoch 2/2  
186/186 - 24s - loss: 0.6809 - accuracy: 0.7076 - 24s/epoch - 129ms/step  
47/47 - 1s - loss: 0.7555 - accuracy: 0.6799 - 737ms/epoch - 16ms/step  
WARNING:tensorflow:Layer lstm\_29 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
Epoch 1/2  
186/186 - 36s - loss: 0.8497 - accuracy: 0.6370 - 36s/epoch - 192ms/step  
Epoch 2/2  
186/186 - 26s - loss: 0.6874 - accuracy: 0.7052 - 26s/epoch - 139ms/step  
47/47 - 1s - loss: 0.7363 - accuracy: 0.6889 - 748ms/epoch - 16ms/step  
WARNING:tensorflow:Layer lstm\_30 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
Epoch 1/2  
186/186 - 37s - loss: 0.8370 - accuracy: 0.6371 - 37s/epoch - 198ms/step  
Epoch 2/2  
186/186 - 26s - loss: 0.6795 - accuracy: 0.7098 - 26s/epoch - 140ms/step  
47/47 - 1s - loss: 0.7777 - accuracy: 0.6652 - 730ms/epoch - 16ms/step  
WARNING:tensorflow:Layer lstm\_31 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as fallback when running on GPU.  
Epoch 1/2  
465/465 - 74s - loss: 0.8138 - accuracy: 0.6524 - 74s/epoch - 159ms/step  
Epoch 2/2  
465/465 - 62s - loss: 0.6739 - accuracy: 0.7108 - 62s/epoch - 134ms/step  
Best: 0.681371 using {'batch\_size': 20, 'epochs': 2}