HAR

September 19, 2020

[172]:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[173]:

```
import os
os.chdir('/content/drive/My Drive/HAR')
! ls -l
```

```
total 2714
-rw------- 1 root root  305776 Sep 19 14:56 best.hdf5
-rw------- 1 root root 1277283 Sep 19 09:48 HAR_EDA.ipynb
-rw------- 1 root root  130898 Sep 19 15:25 HAR_LSTM.ipynb
-rw------- 1 root root  303733 Sep 19 09:51 HAR_PREDICTION_MODELS.ipynb
-rw------- 1 root root  143922 Aug 23  2018 t-sne_perp_10_iter_1000.png
-rw------- 1 root root  135463 Aug 23  2018 t-sne_perp_20_iter_1000.png
-rw------- 1 root root  167195 Aug 23  2018 t-sne_perp_2_iter_1000.png
-rw------- 1 root root  150494 Aug 23  2018 t-sne_perp_50_iter_1000.png
-rw------- 1 root root  157603 Aug 23  2018 t-sne_perp_5_iter_1000.png
drwx------ 5 root root    4096 Sep 18 19:03 UCI_HAR_Dataset
```

[175]:

```
import pandas as pd
import numpy as np
import seaborn as sns
```

[232]:

```
# Activities are the class labels
# It is a 6 class classification
ACTIVITIES = {
    0: 'WALKING',
    1: 'WALKING_UPSTAIRS',
    2: 'WALKING_DOWNSTAIRS',
    3: 'SITTING',
    4: 'STANDING',
    5: 'LAYING',
}
```

```
# Utility function to print the confusion matrix
def confusion_matrix(Y_true, Y_pred):
    Y_true = pd.Series([ACTIVITIES[y] for y in np.argmax(Y_true, axis=1)])
    Y_pred = pd.Series([ACTIVITIES[y] for y in np.argmax(Y_pred, axis=1)])

    return pd.crosstab(Y_true, Y_pred, rownames=['True'], colnames=['Pred'])
```

0.0.1 Data

[177]:

```
# Data directory
DATADIR = 'UCI_HAR_Dataset'
```

[178]:

```
# Raw data signals
# Signals are from Accelerometer and Gyroscope
# The signals are in x,y,z directions
# Sensor signals are filtered to have only body acceleration
# excluding the acceleration due to gravity
# Triaxial acceleration from the accelerometer is total acceleration
SIGNALS = [
    "body_acc_x",
    "body_acc_y",
    "body_acc_z",
    "body_gyro_x",
    "body_gyro_y",
    "body_gyro_z",
    "total_acc_x",
    "total_acc_y",
    "total_acc_z"
]
```

[179]:

```
# Utility function to read the data from csv file
def _read_csv(filename):
    return pd.read_csv(filename, delim_whitespace=True, header=None)

# Utility function to load the load
def load_signals(subset):
    signals_data = []

    for signal in SIGNALS:
        filename = f'UCI_HAR_Dataset/{subset}/Inertial Signals/
 ↪{signal}_{subset}.txt'
        signals_data.append(
            _read_csv(filename).values
        )

    # Transpose is used to change the dimensionality of the output,
    # aggregating the signals by combination of sample/timestep.
    # Resultant shape is (7352 train/2947 test samples, 128 timesteps, 9␣
 ↪signals)
    return np.transpose(signals_data, (1, 2, 0))
```

[180]:
```
def load_y(subset):
    """
    The objective that we are trying to predict is a integer, from 1 to 6,
    that represents a human activity. We return a binary representation of
    every sample objective as a 6 bits vector using One Hot Encoding
    (https://pandas.pydata.org/pandas-docs/stable/generated/pandas.get_dummies.
 ↪html)
    """
    filename = f'UCI_HAR_Dataset/{subset}/y_{subset}.txt'
    y = _read_csv(filename)[0]

    return pd.get_dummies(y).values
```

[181]:
```
def load_data():
    """
    Obtain the dataset from multiple files.
    Returns: X_train, X_test, y_train, y_test
    """
    X_train, X_test = load_signals('train'), load_signals('test')
    y_train, y_test = load_y('train'), load_y('test')

    return X_train, X_test, y_train, y_test
```

[182]:
```
# Importing tensorflow
np.random.seed(42)
import tensorflow as tf
tf.random.set_seed(42)
```

[183]:
```
# Configuring a session
session_conf =  tf.compat.v1.ConfigProto(
    intra_op_parallelism_threads=1,
    inter_op_parallelism_threads=1
)
```

[184]:
```
# Import Keras
from keras import backend as K
sess = tf.compat.v1.Session(graph=tf.compat.v1.get_default_graph(),␣
 ↪config=session_conf)
tf.compat.v1.keras.backend.set_session
```

[184]:
```
<function tensorflow.python.keras.backend.set_session>
```

[185]:
```
# Importing libraries
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers.core import Dense, Dropout
```

[186]:

```
# Initializing parameters
epochs = 30
batch_size = 16
n_hidden = 32
```

[187]:

```
# Utility function to count the number of classes
def _count_classes(y):
    return len(set([tuple(category) for category in y]))
```

[188]:

```
# Loading the train and test data
X_train, X_test, Y_train, Y_test = load_data()
```

[191]:

```
timesteps = len(X_train[0])
input_dim = len(X_train[0][0])
n_classes = _count_classes(Y_train)

print(timesteps)
print(input_dim)
print(len(X_train))
```

```
128
9
7352
```

- Defining the Architecture of LSTM

[192]:

```
import tensorflow as tf
from tensorflow.keras.layers import Dense,Dropout,LSTM,BatchNormalization
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding
```

[199]:

```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(8, input_shape=(timesteps, input_dim),return_sequences= True))
# Adding a dropout layer
model.add(Dropout(0.7))
model.add(BatchNormalization())

model.add(LSTM(16))
model.add(Dropout(0.7))
model.add(BatchNormalization())

# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()

# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

```
Model: "sequential_34"
-----------------------------------------------------------------
Layer (type)                 Output Shape              Param #
=================================================================
lstm_50 (LSTM)               (None, 128, 8)            576
-----------------------------------------------------------------
dropout_43 (Dropout)         (None, 128, 8)            0
-----------------------------------------------------------------
batch_normalization_14 (Batc (None, 128, 8)            32
-----------------------------------------------------------------
lstm_51 (LSTM)               (None, 16)                1600
-----------------------------------------------------------------
dropout_44 (Dropout)         (None, 16)                0
-----------------------------------------------------------------
batch_normalization_15 (Batc (None, 16)                64
-----------------------------------------------------------------
dense_23 (Dense)             (None, 6)                 102
=================================================================
Total params: 2,374
Trainable params: 2,326
Non-trainable params: 48
-----------------------------------------------------------------
```

[200]:

```
filepath = "model.hdf5"

# Keep only a single checkpoint, the best over test accuracy.
checkpoint = tf.keras.callbacks.ModelCheckpoint(filepath,
                            monitor='val_accuracy',
                            verbose=1,
                            save_best_only=True,
                            mode='auto')
```

[201]:

```
# Training the model
history=model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_split=0.33,
          epochs=epochs,
          callbacks=[checkpoint])
```

```
Epoch 1/30
308/308 [==============================] - ETA: 0s - loss: 1.6768 - accuracy:
0.3163
Epoch 00001: val_accuracy improved from -inf to 0.54223, saving model to
```

```
model.hdf5
308/308 [==============================] - 31s 99ms/step - loss: 1.6768 -
accuracy: 0.3163 - val_loss: 1.3072 - val_accuracy: 0.5422
Epoch 2/30
308/308 [==============================] - ETA: 0s - loss: 1.2968 - accuracy:
0.4861
Epoch 00002: val_accuracy improved from 0.54223 to 0.62176, saving model to
model.hdf5
308/308 [==============================] - 31s 99ms/step - loss: 1.2968 -
accuracy: 0.4861 - val_loss: 0.9919 - val_accuracy: 0.6218
Epoch 3/30
308/308 [==============================] - ETA: 0s - loss: 1.1214 - accuracy:
0.5283
Epoch 00003: val_accuracy improved from 0.62176 to 0.63988, saving model to
model.hdf5
308/308 [==============================] - 30s 99ms/step - loss: 1.1214 -
accuracy: 0.5283 - val_loss: 0.8946 - val_accuracy: 0.6399
Epoch 4/30
308/308 [==============================] - ETA: 0s - loss: 1.0098 - accuracy:
0.5446
Epoch 00004: val_accuracy improved from 0.63988 to 0.64153, saving model to
model.hdf5
308/308 [==============================] - 33s 106ms/step - loss: 1.0098 -
accuracy: 0.5446 - val_loss: 0.8524 - val_accuracy: 0.6415
Epoch 5/30
308/308 [==============================] - ETA: 0s - loss: 0.9724 - accuracy:
0.5419
Epoch 00005: val_accuracy did not improve from 0.64153
308/308 [==============================] - 31s 101ms/step - loss: 0.9724 -
accuracy: 0.5419 - val_loss: 0.8268 - val_accuracy: 0.6143
Epoch 6/30
308/308 [==============================] - ETA: 0s - loss: 0.9260 - accuracy:
0.5533
Epoch 00006: val_accuracy improved from 0.64153 to 0.66419, saving model to
model.hdf5
308/308 [==============================] - 31s 100ms/step - loss: 0.9260 -
accuracy: 0.5533 - val_loss: 0.7903 - val_accuracy: 0.6642
Epoch 7/30
308/308 [==============================] - ETA: 0s - loss: 0.9015 - accuracy:
0.5827
Epoch 00007: val_accuracy did not improve from 0.66419
308/308 [==============================] - 30s 98ms/step - loss: 0.9015 -
accuracy: 0.5827 - val_loss: 0.7443 - val_accuracy: 0.6337
Epoch 8/30
308/308 [==============================] - ETA: 0s - loss: 0.8700 - accuracy:
0.5982
Epoch 00008: val_accuracy improved from 0.66419 to 0.67326, saving model to
model.hdf5
```

```
308/308 [==============================] - 30s 98ms/step - loss: 0.8700 -
accuracy: 0.5982 - val_loss: 0.6943 - val_accuracy: 0.6733
Epoch 9/30
308/308 [==============================] - ETA: 0s - loss: 0.8704 - accuracy:
0.5988
Epoch 00009: val_accuracy improved from 0.67326 to 0.68274, saving model to
model.hdf5
308/308 [==============================] - 30s 99ms/step - loss: 0.8704 -
accuracy: 0.5988 - val_loss: 0.6320 - val_accuracy: 0.6827
Epoch 10/30
308/308 [==============================] - ETA: 0s - loss: 0.8428 - accuracy:
0.6250
Epoch 00010: val_accuracy improved from 0.68274 to 0.69716, saving model to
model.hdf5
308/308 [==============================] - 30s 99ms/step - loss: 0.8428 -
accuracy: 0.6250 - val_loss: 0.6347 - val_accuracy: 0.6972
Epoch 11/30
308/308 [==============================] - ETA: 0s - loss: 0.8352 - accuracy:
0.6236
Epoch 00011: val_accuracy did not improve from 0.69716
308/308 [==============================] - 30s 98ms/step - loss: 0.8352 -
accuracy: 0.6236 - val_loss: 0.6819 - val_accuracy: 0.6852
Epoch 12/30
308/308 [==============================] - ETA: 0s - loss: 0.8360 - accuracy:
0.6301
Epoch 00012: val_accuracy improved from 0.69716 to 0.73795, saving model to
model.hdf5
308/308 [==============================] - 30s 98ms/step - loss: 0.8360 -
accuracy: 0.6301 - val_loss: 0.5918 - val_accuracy: 0.7379
Epoch 13/30
308/308 [==============================] - ETA: 0s - loss: 0.8232 - accuracy:
0.6337
Epoch 00013: val_accuracy improved from 0.73795 to 0.77585, saving model to
model.hdf5
308/308 [==============================] - 31s 100ms/step - loss: 0.8232 -
accuracy: 0.6337 - val_loss: 0.5656 - val_accuracy: 0.7759
Epoch 14/30
308/308 [==============================] - ETA: 0s - loss: 0.8075 - accuracy:
0.6414
Epoch 00014: val_accuracy improved from 0.77585 to 0.78863, saving model to
model.hdf5
308/308 [==============================] - 33s 106ms/step - loss: 0.8075 -
accuracy: 0.6414 - val_loss: 0.5536 - val_accuracy: 0.7886
Epoch 15/30
308/308 [==============================] - ETA: 0s - loss: 0.7959 - accuracy:
0.6526
Epoch 00015: val_accuracy did not improve from 0.78863
308/308 [==============================] - 33s 107ms/step - loss: 0.7959 -
```

```
accuracy: 0.6526 - val_loss: 0.5597 - val_accuracy: 0.7833
Epoch 16/30
308/308 [==============================] - ETA: 0s - loss: 0.7657 - accuracy:
0.6623
Epoch 00016: val_accuracy did not improve from 0.78863
308/308 [==============================] - 30s 97ms/step - loss: 0.7657 -
accuracy: 0.6623 - val_loss: 0.5681 - val_accuracy: 0.7474
Epoch 17/30
308/308 [==============================] - ETA: 0s - loss: 0.7720 - accuracy:
0.6642
Epoch 00017: val_accuracy did not improve from 0.78863
308/308 [==============================] - 30s 98ms/step - loss: 0.7720 -
accuracy: 0.6642 - val_loss: 0.5281 - val_accuracy: 0.7606
Epoch 18/30
308/308 [==============================] - ETA: 0s - loss: 0.7522 - accuracy:
0.6788
Epoch 00018: val_accuracy did not improve from 0.78863
308/308 [==============================] - 30s 98ms/step - loss: 0.7522 -
accuracy: 0.6788 - val_loss: 0.5360 - val_accuracy: 0.7651
Epoch 19/30
308/308 [==============================] - ETA: 0s - loss: 0.7411 - accuracy:
0.6737
Epoch 00019: val_accuracy did not improve from 0.78863
308/308 [==============================] - 30s 98ms/step - loss: 0.7411 -
accuracy: 0.6737 - val_loss: 0.5999 - val_accuracy: 0.7598
Epoch 20/30
308/308 [==============================] - ETA: 0s - loss: 0.7169 - accuracy:
0.6902
Epoch 00020: val_accuracy did not improve from 0.78863
308/308 [==============================] - 30s 98ms/step - loss: 0.7169 -
accuracy: 0.6902 - val_loss: 0.6227 - val_accuracy: 0.7557
Epoch 21/30
308/308 [==============================] - ETA: 0s - loss: 0.7180 - accuracy:
0.6920
Epoch 00021: val_accuracy did not improve from 0.78863
308/308 [==============================] - 31s 99ms/step - loss: 0.7180 -
accuracy: 0.6920 - val_loss: 0.6176 - val_accuracy: 0.7602
Epoch 22/30
308/308 [==============================] - ETA: 0s - loss: 0.6963 - accuracy:
0.7054
Epoch 00022: val_accuracy did not improve from 0.78863
308/308 [==============================] - 31s 100ms/step - loss: 0.6963 -
accuracy: 0.7054 - val_loss: 0.6228 - val_accuracy: 0.7672
Epoch 23/30
308/308 [==============================] - ETA: 0s - loss: 0.6889 - accuracy:
0.7038
Epoch 00023: val_accuracy did not improve from 0.78863
308/308 [==============================] - 31s 100ms/step - loss: 0.6889 -
```

```
accuracy: 0.7038 - val_loss: 0.5872 - val_accuracy: 0.7721
Epoch 24/30
308/308 [==============================] - ETA: 0s - loss: 0.6824 - accuracy:
0.7021
Epoch 00024: val_accuracy did not improve from 0.78863
308/308 [==============================] - 31s 100ms/step - loss: 0.6824 -
accuracy: 0.7021 - val_loss: 0.5682 - val_accuracy: 0.7787
Epoch 25/30
308/308 [==============================] - ETA: 0s - loss: 0.6696 - accuracy:
0.7036
Epoch 00025: val_accuracy did not improve from 0.78863
308/308 [==============================] - 35s 114ms/step - loss: 0.6696 -
accuracy: 0.7036 - val_loss: 0.4592 - val_accuracy: 0.7837
Epoch 26/30
308/308 [==============================] - ETA: 0s - loss: 0.6535 - accuracy:
0.7137
Epoch 00026: val_accuracy did not improve from 0.78863
308/308 [==============================] - 30s 98ms/step - loss: 0.6535 -
accuracy: 0.7137 - val_loss: 0.5721 - val_accuracy: 0.7701
Epoch 27/30
308/308 [==============================] - ETA: 0s - loss: 0.6414 - accuracy:
0.7084
Epoch 00027: val_accuracy did not improve from 0.78863
308/308 [==============================] - 30s 99ms/step - loss: 0.6414 -
accuracy: 0.7084 - val_loss: 0.6434 - val_accuracy: 0.7594
Epoch 28/30
308/308 [==============================] - ETA: 0s - loss: 0.6416 - accuracy:
0.7117
Epoch 00028: val_accuracy did not improve from 0.78863
308/308 [==============================] - 31s 100ms/step - loss: 0.6416 -
accuracy: 0.7117 - val_loss: 0.6363 - val_accuracy: 0.7499
Epoch 29/30
308/308 [==============================] - ETA: 0s - loss: 0.6213 - accuracy:
0.7230
Epoch 00029: val_accuracy did not improve from 0.78863
308/308 [==============================] - 31s 99ms/step - loss: 0.6213 -
accuracy: 0.7230 - val_loss: 0.6914 - val_accuracy: 0.7581
Epoch 30/30
308/308 [==============================] - ETA: 0s - loss: 0.6296 - accuracy:
0.7094
Epoch 00030: val_accuracy did not improve from 0.78863
308/308 [==============================] - 31s 100ms/step - loss: 0.6296 -
accuracy: 0.7094 - val_loss: 0.7828 - val_accuracy: 0.7157
```

[233]:

```
# Confusion Matrix

print(confusion_matrix(Y_test, model.predict(X_test)))
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_UPSTAIRS
True
LAYING                 521        0         0        0                16
SITTING                  1      427        60        1                 2
STANDING                 0      128       403        1                 0
WALKING                  2        1       111      351                31
WALKING_DOWNSTAIRS       1        0         0       18               401
WALKING_UPSTAIRS         4        0         5      101               361
```

[236]:

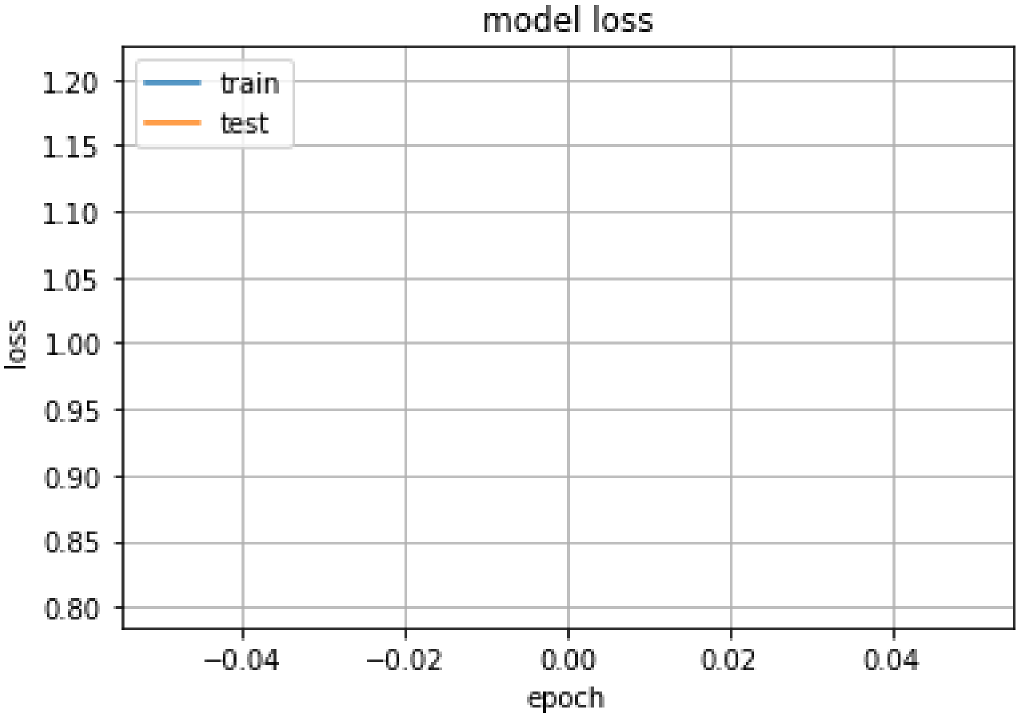
```
score = model.evaluate(X_test, Y_test)
print('\n')
print('-'*100)
print('\n')
print(f'model accuracy is {score[1]}')


import matplotlib.pyplot  as plt
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.grid()
plt.show()
```

```
93/93 [==============================] - 2s 19ms/step - loss: 0.7626 - accuracy:
0.7000


--------------------------------------------------------------------------------
--------------------


model accuracy is 0.7000339031219482
```



[120]:
```
# Initiliazing the sequential model
model2 = Sequential()
# Configuring the parameters
model2.add(LSTM(16, input_shape=(timesteps, input_dim),return_sequences= True))
# Adding a dropout layer
model.add(Dropout(0.3))

model2.add(LSTM(32))
model2.add(Dropout(0.7))

# Adding a dense output layer with sigmoid activation
model2.add(Dense(n_classes, activation='sigmoid'))
model2.summary()

# Compiling the model
model2.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

```
Model: "sequential_21"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_26 (LSTM)               (None, 128, 16)           1664
```

```
-----------------------------------------------------------------
lstm_27 (LSTM)               (None, 32)                6272
-----------------------------------------------------------------
dropout_20 (Dropout)         (None, 32)                0
-----------------------------------------------------------------
dense_12 (Dense)             (None, 6)                 198
=================================================================
Total params: 8,134
Trainable params: 8,134
Non-trainable params: 0
-----------------------------------------------------------------
```

[121]:

```
# Training the model
history2=model2.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=epochs)
```

```
Epoch 1/30
460/460 [==============================] - 47s 101ms/step - loss: 1.3230 -
accuracy: 0.4544 - val_loss: 1.0300 - val_accuracy: 0.5745
Epoch 2/30
460/460 [==============================] - 46s 100ms/step - loss: 0.9668 -
accuracy: 0.5934 - val_loss: 0.8456 - val_accuracy: 0.6661
Epoch 3/30
460/460 [==============================] - 46s 100ms/step - loss: 0.8053 -
accuracy: 0.6620 - val_loss: 0.9324 - val_accuracy: 0.6376
Epoch 4/30
460/460 [==============================] - 46s 99ms/step - loss: 0.7820 -
accuracy: 0.6748 - val_loss: 0.7659 - val_accuracy: 0.6851
Epoch 5/30
460/460 [==============================] - 46s 99ms/step - loss: 0.7270 -
accuracy: 0.6982 - val_loss: 0.6875 - val_accuracy: 0.6973
Epoch 6/30
460/460 [==============================] - 46s 100ms/step - loss: 0.6719 -
accuracy: 0.7274 - val_loss: 0.6823 - val_accuracy: 0.7201
Epoch 7/30
460/460 [==============================] - 48s 103ms/step - loss: 0.5972 -
accuracy: 0.7497 - val_loss: 0.7390 - val_accuracy: 0.7197
Epoch 8/30
460/460 [==============================] - 46s 101ms/step - loss: 0.5600 -
accuracy: 0.7726 - val_loss: 0.7427 - val_accuracy: 0.7384
Epoch 9/30
460/460 [==============================] - 47s 103ms/step - loss: 0.5384 -
accuracy: 0.7904 - val_loss: 0.7459 - val_accuracy: 0.7479
Epoch 10/30
```

```
460/460 [==============================] - 46s 99ms/step - loss: 0.4892 -
accuracy: 0.8017 - val_loss: 0.6925 - val_accuracy: 0.7333
Epoch 11/30
460/460 [==============================] - 46s 99ms/step - loss: 0.4384 -
accuracy: 0.8259 - val_loss: 0.7154 - val_accuracy: 0.8107
Epoch 12/30
460/460 [==============================] - 46s 99ms/step - loss: 0.4056 -
accuracy: 0.8716 - val_loss: 0.6425 - val_accuracy: 0.8137
Epoch 13/30
460/460 [==============================] - 46s 99ms/step - loss: 0.3559 -
accuracy: 0.8953 - val_loss: 0.5955 - val_accuracy: 0.8500
Epoch 14/30
460/460 [==============================] - 47s 101ms/step - loss: 0.3242 -
accuracy: 0.9109 - val_loss: 0.5967 - val_accuracy: 0.8714
Epoch 15/30
460/460 [==============================] - 46s 99ms/step - loss: 0.3070 -
accuracy: 0.9181 - val_loss: 0.4692 - val_accuracy: 0.8816
Epoch 16/30
460/460 [==============================] - 49s 106ms/step - loss: 0.2891 -
accuracy: 0.9208 - val_loss: 0.5342 - val_accuracy: 0.8887
Epoch 17/30
460/460 [==============================] - 45s 98ms/step - loss: 0.2618 -
accuracy: 0.9257 - val_loss: 0.6217 - val_accuracy: 0.8643
Epoch 18/30
460/460 [==============================] - 45s 98ms/step - loss: 0.2587 -
accuracy: 0.9232 - val_loss: 0.5052 - val_accuracy: 0.8850
Epoch 19/30
460/460 [==============================] - 45s 97ms/step - loss: 0.2679 -
accuracy: 0.9279 - val_loss: 0.5180 - val_accuracy: 0.8765
Epoch 20/30
460/460 [==============================] - 46s 99ms/step - loss: 0.2504 -
accuracy: 0.9285 - val_loss: 0.4084 - val_accuracy: 0.8918
Epoch 21/30
460/460 [==============================] - 45s 99ms/step - loss: 0.2336 -
accuracy: 0.9346 - val_loss: 0.4596 - val_accuracy: 0.8972
Epoch 22/30
460/460 [==============================] - 45s 97ms/step - loss: 0.2341 -
accuracy: 0.9316 - val_loss: 0.4568 - val_accuracy: 0.8972
Epoch 23/30
460/460 [==============================] - 48s 104ms/step - loss: 0.2328 -
accuracy: 0.9366 - val_loss: 0.4981 - val_accuracy: 0.8951
Epoch 24/30
460/460 [==============================] - 46s 100ms/step - loss: 0.2197 -
accuracy: 0.9347 - val_loss: 0.6157 - val_accuracy: 0.8955
Epoch 25/30
460/460 [==============================] - 45s 98ms/step - loss: 0.2079 -
accuracy: 0.9347 - val_loss: 0.5205 - val_accuracy: 0.8846
Epoch 26/30
```

```
460/460 [==============================] - 45s 97ms/step - loss: 0.2066 -
accuracy: 0.9354 - val_loss: 0.5888 - val_accuracy: 0.9030
Epoch 27/30
460/460 [==============================] - 46s 99ms/step - loss: 0.2162 -
accuracy: 0.9347 - val_loss: 0.5928 - val_accuracy: 0.8870
Epoch 28/30
460/460 [==============================] - 45s 97ms/step - loss: 0.1932 -
accuracy: 0.9403 - val_loss: 0.5316 - val_accuracy: 0.8850
Epoch 29/30
460/460 [==============================] - 46s 100ms/step - loss: 0.2149 -
accuracy: 0.9358 - val_loss: 0.5272 - val_accuracy: 0.8867
Epoch 30/30
460/460 [==============================] - 46s 101ms/step - loss: 0.1912 -
accuracy: 0.9385 - val_loss: 0.4637 - val_accuracy: 0.9091
```

[122]:

```
# Confusion Matrix
print(confusion_matrix(Y_test, model2.predict(X_test)))
```

```
Pred                LAYING  SITTING  ...  WALKING_DOWNSTAIRS  WALKING_UPSTAIRS
True                                 ...
LAYING                 510        0  ...                   0                27
SITTING                  0      393  ...                   2                 3
STANDING                 0       52  ...                   0                 0
WALKING                  0        0  ...                   0                16
WALKING_DOWNSTAIRS       0        0  ...                 397                22
WALKING_UPSTAIRS         0        0  ...                   5               424

[6 rows x 6 columns]
```

[123]:

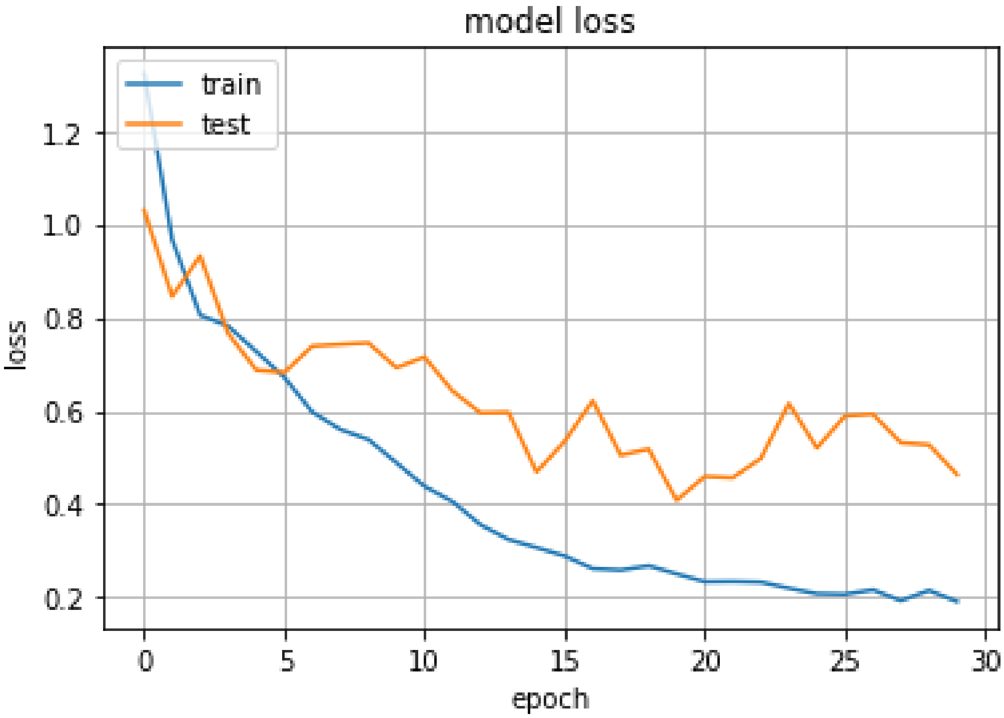
```
score = model2.evaluate(X_test, Y_test)
print('\n')
print('-'*100)
print('\n')
print(f'model accuracy is {score[1]}')
```

```
93/93 [==============================] - 2s 20ms/step - loss: 0.4637 - accuracy:
0.9091


--------------------------------------------------------------------------------
--------------------


model accuracy is 0.9090600609779358
```

[124]:
```
import matplotlib.pyplot  as plt
plt.plot(history2.history['loss'])
plt.plot(history2.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.grid()
plt.show()
```



[226]:
```
# Initiliazing the sequential model
model3 = Sequential()
# Configuring the parameters
model3.add(LSTM(32, input_shape=(timesteps, input_dim),return_sequences= True))
# Adding a dropout layer
model3.add(Dropout(0.7))
model3.add(BatchNormalization())

model3.add(LSTM(64))
model3.add(Dropout(0.7))
model3.add(BatchNormalization())

# Adding a dense output layer with sigmoid activation
model3.add(Dense(n_classes, activation='sigmoid'))
```

```
model3.summary()

# Compiling the model
model3.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

```
Model: "sequential_38"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_58 (LSTM)               (None, 128, 32)           5376
_________________________________________________________________
dropout_51 (Dropout)         (None, 128, 32)           0
_________________________________________________________________
batch_normalization_22 (Batc (None, 128, 32)           128
_________________________________________________________________
lstm_59 (LSTM)               (None, 64)                24832
_________________________________________________________________
dropout_52 (Dropout)         (None, 64)                0
_________________________________________________________________
batch_normalization_23 (Batc (None, 64)                256
_________________________________________________________________
dense_27 (Dense)             (None, 6)                 390
=================================================================
Total params: 30,982
Trainable params: 30,790
Non-trainable params: 192
_________________________________________________________________
```

[237]:

```
# Training the model
history3=model3.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_split=0.33,
          epochs=epochs)
```

```
Epoch 1/30
308/308 [==============================] - ETA: 0s - loss: 0.8143 - accuracy:
0.6370
Epoch 00001: val_accuracy did not improve from 0.67079
308/308 [==============================] - 39s 125ms/step - loss: 0.8143 -
accuracy: 0.6370 - val_loss: 0.9845 - val_accuracy: 0.6296
Epoch 2/30
308/308 [==============================] - ETA: 0s - loss: 0.7174 - accuracy:
0.6662
```

```
Epoch 00002: val_accuracy improved from 0.67079 to 0.77585, saving model to
model3.hdf5
308/308 [==============================] - 38s 123ms/step - loss: 0.7174 -
accuracy: 0.6662 - val_loss: 0.6756 - val_accuracy: 0.7759
Epoch 3/30
308/308 [==============================] - ETA: 0s - loss: 0.6431 - accuracy:
0.7062
Epoch 00003: val_accuracy did not improve from 0.77585
308/308 [==============================] - 40s 129ms/step - loss: 0.6431 -
accuracy: 0.7062 - val_loss: 0.7072 - val_accuracy: 0.7425
Epoch 4/30
308/308 [==============================] - ETA: 0s - loss: 0.5504 - accuracy:
0.7773
Epoch 00004: val_accuracy improved from 0.77585 to 0.79687, saving model to
model3.hdf5
308/308 [==============================] - 38s 124ms/step - loss: 0.5504 -
accuracy: 0.7773 - val_loss: 0.5156 - val_accuracy: 0.7969
Epoch 5/30
308/308 [==============================] - ETA: 0s - loss: 0.3974 - accuracy:
0.8623
Epoch 00005: val_accuracy improved from 0.79687 to 0.89699, saving model to
model3.hdf5
308/308 [==============================] - 38s 124ms/step - loss: 0.3974 -
accuracy: 0.8623 - val_loss: 0.4164 - val_accuracy: 0.8970
Epoch 6/30
308/308 [==============================] - ETA: 0s - loss: 0.3217 - accuracy:
0.8979
Epoch 00006: val_accuracy improved from 0.89699 to 0.92213, saving model to
model3.hdf5
308/308 [==============================] - 39s 126ms/step - loss: 0.3217 -
accuracy: 0.8979 - val_loss: 0.2948 - val_accuracy: 0.9221
Epoch 7/30
308/308 [==============================] - ETA: 0s - loss: 0.2920 - accuracy:
0.9113
Epoch 00007: val_accuracy did not improve from 0.92213
308/308 [==============================] - 39s 125ms/step - loss: 0.2920 -
accuracy: 0.9113 - val_loss: 0.3887 - val_accuracy: 0.9015
Epoch 8/30
308/308 [==============================] - ETA: 0s - loss: 0.2515 - accuracy:
0.9178
Epoch 00008: val_accuracy did not improve from 0.92213
308/308 [==============================] - 38s 124ms/step - loss: 0.2515 -
accuracy: 0.9178 - val_loss: 0.3164 - val_accuracy: 0.9184
Epoch 9/30
308/308 [==============================] - ETA: 0s - loss: 0.2405 - accuracy:
0.9204
Epoch 00009: val_accuracy improved from 0.92213 to 0.92872, saving model to
model3.hdf5
```

```
308/308 [==============================] - 39s 126ms/step - loss: 0.2405 -
accuracy: 0.9204 - val_loss: 0.2845 - val_accuracy: 0.9287
Epoch 10/30
308/308 [==============================] - ETA: 0s - loss: 0.2249 - accuracy:
0.9245
Epoch 00010: val_accuracy did not improve from 0.92872
308/308 [==============================] - 37s 122ms/step - loss: 0.2249 -
accuracy: 0.9245 - val_loss: 0.5018 - val_accuracy: 0.8916
Epoch 11/30
308/308 [==============================] - ETA: 0s - loss: 0.2165 - accuracy:
0.9249
Epoch 00011: val_accuracy did not improve from 0.92872
308/308 [==============================] - 39s 128ms/step - loss: 0.2165 -
accuracy: 0.9249 - val_loss: 0.2805 - val_accuracy: 0.9069
Epoch 12/30
308/308 [==============================] - ETA: 0s - loss: 0.2003 - accuracy:
0.9342
Epoch 00012: val_accuracy did not improve from 0.92872
308/308 [==============================] - 38s 122ms/step - loss: 0.2003 -
accuracy: 0.9342 - val_loss: 0.2433 - val_accuracy: 0.9271
Epoch 13/30
308/308 [==============================] - ETA: 0s - loss: 0.1964 - accuracy:
0.9310
Epoch 00013: val_accuracy did not improve from 0.92872
308/308 [==============================] - 38s 123ms/step - loss: 0.1964 -
accuracy: 0.9310 - val_loss: 0.3081 - val_accuracy: 0.9279
Epoch 14/30
308/308 [==============================] - ETA: 0s - loss: 0.1862 - accuracy:
0.9304
Epoch 00014: val_accuracy did not improve from 0.92872
308/308 [==============================] - 38s 122ms/step - loss: 0.1862 -
accuracy: 0.9304 - val_loss: 0.3972 - val_accuracy: 0.8842
Epoch 15/30
308/308 [==============================] - ETA: 0s - loss: 0.1907 - accuracy:
0.9342
Epoch 00015: val_accuracy improved from 0.92872 to 0.94396, saving model to
model3.hdf5
308/308 [==============================] - 38s 122ms/step - loss: 0.1907 -
accuracy: 0.9342 - val_loss: 0.2917 - val_accuracy: 0.9440
Epoch 16/30
308/308 [==============================] - ETA: 0s - loss: 0.1858 - accuracy:
0.9342
Epoch 00016: val_accuracy did not improve from 0.94396
308/308 [==============================] - 38s 122ms/step - loss: 0.1858 -
accuracy: 0.9342 - val_loss: 0.3148 - val_accuracy: 0.9192
Epoch 17/30
308/308 [==============================] - ETA: 0s - loss: 0.1851 - accuracy:
0.9348
```

```
Epoch 00017: val_accuracy did not improve from 0.94396
308/308 [==============================] - 39s 125ms/step - loss: 0.1851 -
accuracy: 0.9348 - val_loss: 0.2293 - val_accuracy: 0.9172
Epoch 18/30
308/308 [==============================] - ETA: 0s - loss: 0.1832 - accuracy:
0.9306
Epoch 00018: val_accuracy did not improve from 0.94396
308/308 [==============================] - 37s 122ms/step - loss: 0.1832 -
accuracy: 0.9306 - val_loss: 0.3839 - val_accuracy: 0.9205
Epoch 19/30
308/308 [==============================] - ETA: 0s - loss: 0.1634 - accuracy:
0.9316
Epoch 00019: val_accuracy did not improve from 0.94396
308/308 [==============================] - 38s 123ms/step - loss: 0.1634 -
accuracy: 0.9316 - val_loss: 0.4043 - val_accuracy: 0.9131
Epoch 20/30
308/308 [==============================] - ETA: 0s - loss: 0.1665 - accuracy:
0.9369
Epoch 00020: val_accuracy did not improve from 0.94396
308/308 [==============================] - 39s 127ms/step - loss: 0.1665 -
accuracy: 0.9369 - val_loss: 0.3202 - val_accuracy: 0.9221
Epoch 21/30
308/308 [==============================] - ETA: 0s - loss: 0.1621 - accuracy:
0.9358
Epoch 00021: val_accuracy did not improve from 0.94396
308/308 [==============================] - 38s 122ms/step - loss: 0.1621 -
accuracy: 0.9358 - val_loss: 0.3738 - val_accuracy: 0.9291
Epoch 22/30
308/308 [==============================] - ETA: 0s - loss: 0.1774 - accuracy:
0.9373
Epoch 00022: val_accuracy did not improve from 0.94396
308/308 [==============================] - 37s 121ms/step - loss: 0.1774 -
accuracy: 0.9373 - val_loss: 0.3473 - val_accuracy: 0.9328
Epoch 23/30
308/308 [==============================] - ETA: 0s - loss: 0.1570 - accuracy:
0.9395
Epoch 00023: val_accuracy improved from 0.94396 to 0.94726, saving model to
model3.hdf5
308/308 [==============================] - 38s 123ms/step - loss: 0.1570 -
accuracy: 0.9395 - val_loss: 0.2456 - val_accuracy: 0.9473
Epoch 24/30
308/308 [==============================] - ETA: 0s - loss: 0.1554 - accuracy:
0.9409
Epoch 00024: val_accuracy did not improve from 0.94726
308/308 [==============================] - 38s 122ms/step - loss: 0.1554 -
accuracy: 0.9409 - val_loss: 0.3255 - val_accuracy: 0.9201
Epoch 25/30
308/308 [==============================] - ETA: 0s - loss: 0.1644 - accuracy:
```

```
0.9407
Epoch 00025: val_accuracy did not improve from 0.94726
308/308 [==============================] - 38s 124ms/step - loss: 0.1644 -
accuracy: 0.9407 - val_loss: 0.3134 - val_accuracy: 0.9081
Epoch 26/30
308/308 [==============================] - ETA: 0s - loss: 0.1555 - accuracy:
0.9399
Epoch 00026: val_accuracy did not improve from 0.94726
308/308 [==============================] - 38s 122ms/step - loss: 0.1555 -
accuracy: 0.9399 - val_loss: 0.4808 - val_accuracy: 0.8920
Epoch 27/30
308/308 [==============================] - ETA: 0s - loss: 0.1691 - accuracy:
0.9385
Epoch 00027: val_accuracy did not improve from 0.94726
308/308 [==============================] - 37s 122ms/step - loss: 0.1691 -
accuracy: 0.9385 - val_loss: 0.5830 - val_accuracy: 0.8991
Epoch 28/30
308/308 [==============================] - ETA: 0s - loss: 0.1521 - accuracy:
0.9436
Epoch 00028: val_accuracy did not improve from 0.94726
308/308 [==============================] - 39s 128ms/step - loss: 0.1521 -
accuracy: 0.9436 - val_loss: 0.3583 - val_accuracy: 0.9234
Epoch 29/30
308/308 [==============================] - ETA: 0s - loss: 0.1423 - accuracy:
0.9399
Epoch 00029: val_accuracy did not improve from 0.94726
308/308 [==============================] - 37s 121ms/step - loss: 0.1423 -
accuracy: 0.9399 - val_loss: 0.2860 - val_accuracy: 0.9209
Epoch 30/30
308/308 [==============================] - ETA: 0s - loss: 0.1500 - accuracy:
0.9438
Epoch 00030: val_accuracy did not improve from 0.94726
308/308 [==============================] - 37s 121ms/step - loss: 0.1500 -
accuracy: 0.9438 - val_loss: 0.2755 - val_accuracy: 0.9320
```

[243]:

```
# Confusion Matrix
print(confusion_matrix(Y_test, model3.predict(X_test)))
```

```
Pred                LAYING  SITTING  ...  WALKING_DOWNSTAIRS  WALKING_UPSTAIRS
True                                 ...
LAYING                 517        0  ...                   0                13
SITTING                  2      451  ...                   0                 0
STANDING                 0      125  ...                   0                 0
WALKING                  0        1  ...                   1                 1
WALKING_DOWNSTAIRS       0        0  ...                 388                21
WALKING_UPSTAIRS         0        0  ...                   0               471

[6 rows x 6 columns]
```

[245]:

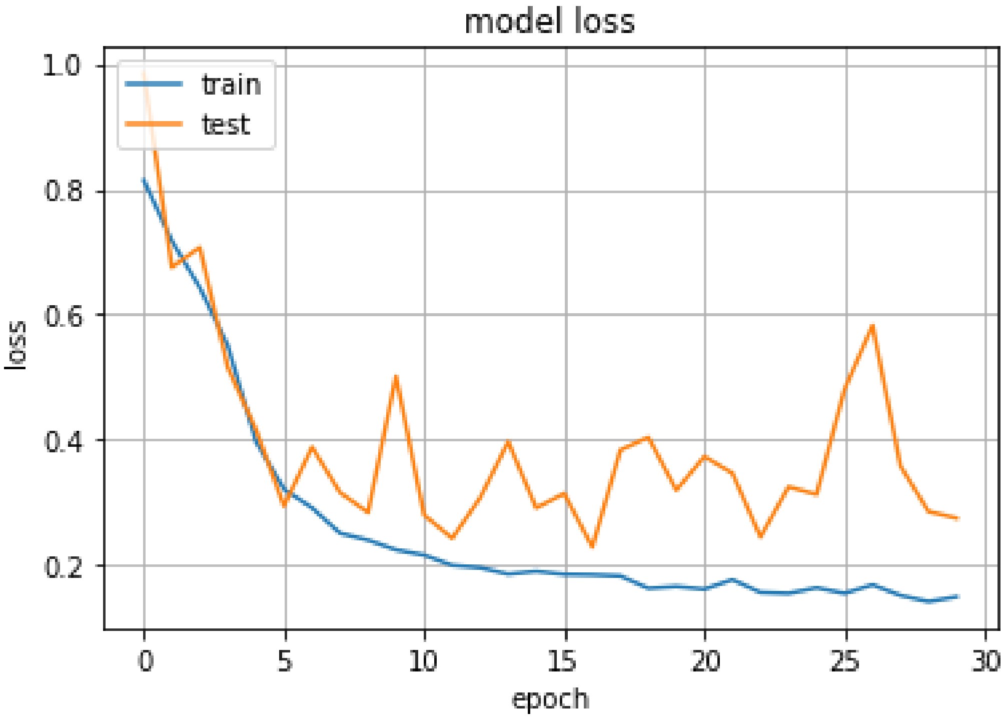
```
score = model3.evaluate(X_test, Y_test)
print('\n')
print('-'*100)
print('\n')
print(f'model accuracy is {score[1]}')

import matplotlib.pyplot  as plt
plt.plot(history3.history['loss'])
plt.plot(history3.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.grid()
plt.show()
```

```
93/93 [==============================] - 3s 34ms/step - loss: 0.3121 - accuracy:
0.9223


--------------------------------------------------------------------------------
--------------------


model accuracy is 0.9222938418388367
```



[267]:

```
# Initiliazing the sequential model
model4 = Sequential()
# Configuring the parameters
model4.add(LSTM(8, input_shape=(timesteps, input_dim),return_sequences= True))
# Adding a dropout layer
model4.add(Dropout(0.3))
model4.add(BatchNormalization())

model4.add(LSTM(16,return_sequences= True))
model4.add(Dropout(0.5))
model4.add(BatchNormalization())

model4.add(LSTM(32))
model4.add(Dropout(0.7))
model4.add(BatchNormalization())

# Adding a dense output layer with sigmoid activation
model4.add(Dense(n_classes, activation='sigmoid'))
model4.summary()

# Compiling the model
model4.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

```
Model: "sequential_45"
-----------------------------------------------------------------
Layer (type)                 Output Shape              Param #
=================================================================
lstm_75 (LSTM)               (None, 128, 8)            576
-----------------------------------------------------------------
dropout_65 (Dropout)         (None, 128, 8)            0
-----------------------------------------------------------------
batch_normalization_36 (Batc (None, 128, 8)            32
-----------------------------------------------------------------
lstm_76 (LSTM)               (None, 128, 16)           1600
-----------------------------------------------------------------
dropout_66 (Dropout)         (None, 128, 16)           0
-----------------------------------------------------------------
batch_normalization_37 (Batc (None, 128, 16)           64
-----------------------------------------------------------------
lstm_77 (LSTM)               (None, 32)                6272
-----------------------------------------------------------------
dropout_67 (Dropout)         (None, 32)                0
-----------------------------------------------------------------
batch_normalization_38 (Batc (None, 32)                128
_________________________________________________________________
dense_31 (Dense)             (None, 6)                 198
=================================================================
Total params: 8,870
Trainable params: 8,758
Non-trainable params: 112
_________________________________________________________________
```

[268]:

```python
filepath = "model4.hdf5"

# Keep only a single checkpoint, the best over test accuracy.
checkpoint = tf.keras.callbacks.ModelCheckpoint(filepath,
                            monitor='val_accuracy',
                            verbose=1,
                            save_best_only=True,
                            mode='auto')

# Training the model
history4=model4.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_split=0.33,
          epochs=epochs,
          callbacks=[checkpoint])
```

```
Epoch 1/30
308/308 [==============================] - ETA: 0s - loss: 1.4209 - accuracy:
0.4386
Epoch 00001: val_accuracy improved from -inf to 0.61846, saving model to
model4.hdf5
308/308 [==============================] - 51s 166ms/step - loss: 1.4209 -
accuracy: 0.4386 - val_loss: 1.0668 - val_accuracy: 0.6185
Epoch 2/30
308/308 [==============================] - ETA: 0s - loss: 1.0284 - accuracy:
0.5523
Epoch 00002: val_accuracy improved from 0.61846 to 0.62835, saving model to
model4.hdf5
308/308 [==============================] - 49s 158ms/step - loss: 1.0284 -
accuracy: 0.5523 - val_loss: 0.8194 - val_accuracy: 0.6283
Epoch 3/30
308/308 [==============================] - ETA: 0s - loss: 0.8877 - accuracy:
0.6234
Epoch 00003: val_accuracy improved from 0.62835 to 0.63947, saving model to
model4.hdf5
308/308 [==============================] - 49s 159ms/step - loss: 0.8877 -
accuracy: 0.6234 - val_loss: 0.7776 - val_accuracy: 0.6395
Epoch 4/30
```

```
308/308 [==============================] - ETA: 0s - loss: 0.7913 - accuracy:
0.6384
Epoch 00004: val_accuracy improved from 0.63947 to 0.73795, saving model to
model4.hdf5
308/308 [==============================] - 49s 159ms/step - loss: 0.7913 -
accuracy: 0.6384 - val_loss: 0.7430 - val_accuracy: 0.7379
Epoch 5/30
308/308 [==============================] - ETA: 0s - loss: 0.7437 - accuracy:
0.6731
Epoch 00005: val_accuracy improved from 0.73795 to 0.83024, saving model to
model4.hdf5
308/308 [==============================] - 48s 154ms/step - loss: 0.7437 -
accuracy: 0.6731 - val_loss: 0.6578 - val_accuracy: 0.8302
Epoch 6/30
308/308 [==============================] - ETA: 0s - loss: 0.6974 - accuracy:
0.7098
Epoch 00006: val_accuracy did not improve from 0.83024
308/308 [==============================] - 47s 153ms/step - loss: 0.6974 -
accuracy: 0.7098 - val_loss: 0.6332 - val_accuracy: 0.7833
Epoch 7/30
308/308 [==============================] - ETA: 0s - loss: 0.6341 - accuracy:
0.7541
Epoch 00007: val_accuracy improved from 0.83024 to 0.84714, saving model to
model4.hdf5
308/308 [==============================] - 48s 155ms/step - loss: 0.6341 -
accuracy: 0.7541 - val_loss: 0.5832 - val_accuracy: 0.8471
Epoch 8/30
308/308 [==============================] - ETA: 0s - loss: 0.5420 - accuracy:
0.8059
Epoch 00008: val_accuracy improved from 0.84714 to 0.89658, saving model to
model4.hdf5
308/308 [==============================] - 52s 168ms/step - loss: 0.5420 -
accuracy: 0.8059 - val_loss: 0.4593 - val_accuracy: 0.8966
Epoch 9/30
308/308 [==============================] - ETA: 0s - loss: 0.5130 - accuracy:
0.8282
Epoch 00009: val_accuracy improved from 0.89658 to 0.90812, saving model to
model4.hdf5
308/308 [==============================] - 48s 155ms/step - loss: 0.5130 -
accuracy: 0.8282 - val_loss: 0.4103 - val_accuracy: 0.9081
Epoch 10/30
308/308 [==============================] - ETA: 0s - loss: 0.4608 - accuracy:
0.8536
Epoch 00010: val_accuracy did not improve from 0.90812
308/308 [==============================] - 48s 155ms/step - loss: 0.4608 -
accuracy: 0.8536 - val_loss: 0.4384 - val_accuracy: 0.8797
Epoch 11/30
308/308 [==============================] - ETA: 0s - loss: 0.4259 - accuracy:
```

```
0.8623
Epoch 00011: val_accuracy did not improve from 0.90812
308/308 [==============================] - 47s 154ms/step - loss: 0.4259 -
accuracy: 0.8623 - val_loss: 0.5169 - val_accuracy: 0.8232
Epoch 12/30
308/308 [==============================] - ETA: 0s - loss: 0.3922 - accuracy:
0.8729
Epoch 00012: val_accuracy improved from 0.90812 to 0.92213, saving model to
model4.hdf5
308/308 [==============================] - 47s 153ms/step - loss: 0.3922 -
accuracy: 0.8729 - val_loss: 0.3394 - val_accuracy: 0.9221
Epoch 13/30
308/308 [==============================] - ETA: 0s - loss: 0.3531 - accuracy:
0.8857
Epoch 00013: val_accuracy improved from 0.92213 to 0.92913, saving model to
model4.hdf5
308/308 [==============================] - 47s 153ms/step - loss: 0.3531 -
accuracy: 0.8857 - val_loss: 0.3259 - val_accuracy: 0.9291
Epoch 14/30
308/308 [==============================] - ETA: 0s - loss: 0.3538 - accuracy:
0.8835
Epoch 00014: val_accuracy did not improve from 0.92913
308/308 [==============================] - 48s 157ms/step - loss: 0.3538 -
accuracy: 0.8835 - val_loss: 0.4094 - val_accuracy: 0.8970
Epoch 15/30
308/308 [==============================] - ETA: 0s - loss: 0.3575 - accuracy:
0.8828
Epoch 00015: val_accuracy did not improve from 0.92913
308/308 [==============================] - 47s 153ms/step - loss: 0.3575 -
accuracy: 0.8828 - val_loss: 0.3555 - val_accuracy: 0.9147
Epoch 16/30
308/308 [==============================] - ETA: 0s - loss: 0.3113 - accuracy:
0.8969
Epoch 00016: val_accuracy did not improve from 0.92913
308/308 [==============================] - 47s 152ms/step - loss: 0.3113 -
accuracy: 0.8969 - val_loss: 0.4032 - val_accuracy: 0.9085
Epoch 17/30
308/308 [==============================] - ETA: 0s - loss: 0.3101 - accuracy:
0.8975
Epoch 00017: val_accuracy did not improve from 0.92913
308/308 [==============================] - 47s 153ms/step - loss: 0.3101 -
accuracy: 0.8975 - val_loss: 0.4618 - val_accuracy: 0.8797
Epoch 18/30
308/308 [==============================] - ETA: 0s - loss: 0.3311 - accuracy:
0.8920
Epoch 00018: val_accuracy did not improve from 0.92913
308/308 [==============================] - 47s 152ms/step - loss: 0.3311 -
accuracy: 0.8920 - val_loss: 0.4455 - val_accuracy: 0.8871
```

```
Epoch 19/30
308/308 [==============================] - ETA: 0s - loss: 0.2739 - accuracy:
0.9052
Epoch 00019: val_accuracy did not improve from 0.92913
308/308 [==============================] - 47s 152ms/step - loss: 0.2739 -
accuracy: 0.9052 - val_loss: 0.4016 - val_accuracy: 0.9180
Epoch 20/30
308/308 [==============================] - ETA: 0s - loss: 0.2805 - accuracy:
0.9038
Epoch 00020: val_accuracy improved from 0.92913 to 0.92954, saving model to
model4.hdf5
308/308 [==============================] - 48s 156ms/step - loss: 0.2805 -
accuracy: 0.9038 - val_loss: 0.3438 - val_accuracy: 0.9295
Epoch 21/30
308/308 [==============================] - ETA: 0s - loss: 0.2862 - accuracy:
0.9048
Epoch 00021: val_accuracy did not improve from 0.92954
308/308 [==============================] - 48s 154ms/step - loss: 0.2862 -
accuracy: 0.9048 - val_loss: 0.4158 - val_accuracy: 0.9168
Epoch 22/30
308/308 [==============================] - ETA: 0s - loss: 0.2928 - accuracy:
0.9040
Epoch 00022: val_accuracy improved from 0.92954 to 0.93325, saving model to
model4.hdf5
308/308 [==============================] - 47s 153ms/step - loss: 0.2928 -
accuracy: 0.9040 - val_loss: 0.3491 - val_accuracy: 0.9333
Epoch 23/30
308/308 [==============================] - ETA: 0s - loss: 0.2722 - accuracy:
0.9050
Epoch 00023: val_accuracy improved from 0.93325 to 0.94149, saving model to
model4.hdf5
308/308 [==============================] - 48s 156ms/step - loss: 0.2722 -
accuracy: 0.9050 - val_loss: 0.3066 - val_accuracy: 0.9415
Epoch 24/30
308/308 [==============================] - ETA: 0s - loss: 0.2757 - accuracy:
0.9121
Epoch 00024: val_accuracy did not improve from 0.94149
308/308 [==============================] - 48s 155ms/step - loss: 0.2757 -
accuracy: 0.9121 - val_loss: 0.4318 - val_accuracy: 0.8859
Epoch 25/30
308/308 [==============================] - ETA: 0s - loss: 0.2660 - accuracy:
0.9078
Epoch 00025: val_accuracy did not improve from 0.94149
308/308 [==============================] - 48s 155ms/step - loss: 0.2660 -
accuracy: 0.9078 - val_loss: 0.3996 - val_accuracy: 0.9209
Epoch 26/30
308/308 [==============================] - ETA: 0s - loss: 0.2700 - accuracy:
0.9072
```

```
Epoch 00026: val_accuracy did not improve from 0.94149
308/308 [==============================] - 47s 153ms/step - loss: 0.2700 -
accuracy: 0.9072 - val_loss: 0.4258 - val_accuracy: 0.8949
Epoch 27/30
308/308 [==============================] - ETA: 0s - loss: 0.2652 - accuracy:
0.9078
Epoch 00027: val_accuracy did not improve from 0.94149
308/308 [==============================] - 49s 160ms/step - loss: 0.2652 -
accuracy: 0.9078 - val_loss: 0.3515 - val_accuracy: 0.9349
Epoch 28/30
308/308 [==============================] - ETA: 0s - loss: 0.2355 - accuracy:
0.9182
Epoch 00028: val_accuracy did not improve from 0.94149
308/308 [==============================] - 47s 154ms/step - loss: 0.2355 -
accuracy: 0.9182 - val_loss: 0.3540 - val_accuracy: 0.9271
Epoch 29/30
308/308 [==============================] - ETA: 0s - loss: 0.2427 - accuracy:
0.9172
Epoch 00029: val_accuracy did not improve from 0.94149
308/308 [==============================] - 48s 157ms/step - loss: 0.2427 -
accuracy: 0.9172 - val_loss: 0.4068 - val_accuracy: 0.9254
Epoch 30/30
308/308 [==============================] - ETA: 0s - loss: 0.2526 - accuracy:
0.9084
Epoch 00030: val_accuracy did not improve from 0.94149
308/308 [==============================] - 48s 156ms/step - loss: 0.2526 -
accuracy: 0.9084 - val_loss: 0.4157 - val_accuracy: 0.9052
```

[269]:

```
# Confusion Matrix
print(confusion_matrix(Y_test, model4.predict(X_test)))
```

```
Pred                LAYING  SITTING  ...  WALKING_DOWNSTAIRS  WALKING_UPSTAIRS
True                                 ...
LAYING                 509        1  ...                   0                26
SITTING                  0      435  ...                   0                 4
STANDING                 0      128  ...                   0                 0
WALKING                  0       16  ...                   1                27
WALKING_DOWNSTAIRS       0        0  ...                 414                 5
WALKING_UPSTAIRS         0        5  ...                  22               429

[6 rows x 6 columns]
```

[270]:

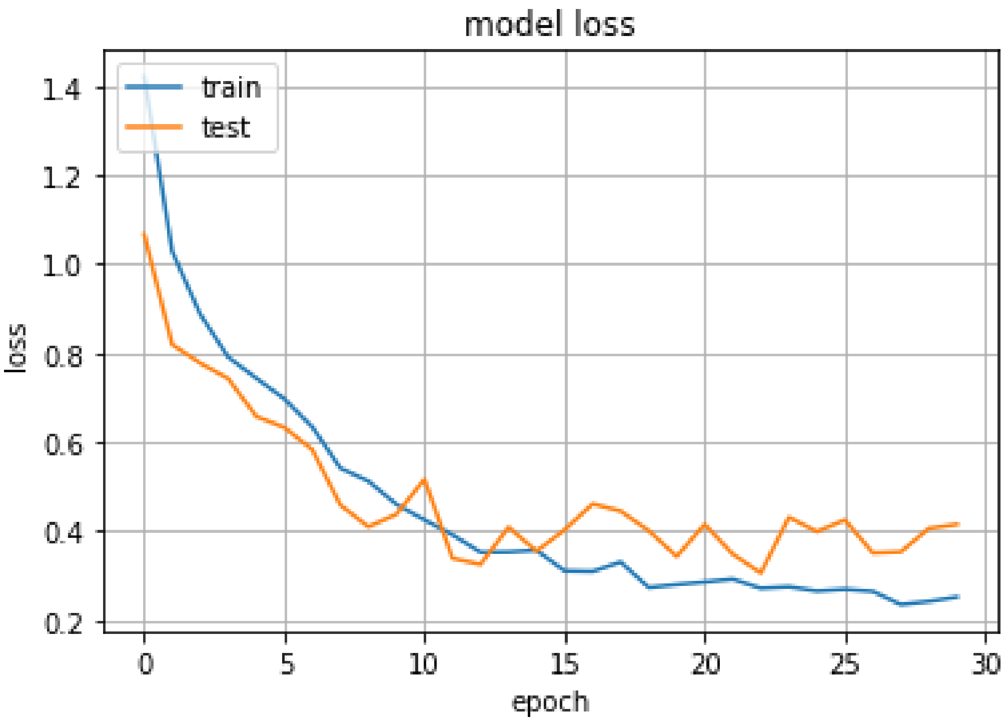
```
score = model4.evaluate(X_test, Y_test)
print('\n')
print('-'*100)
print('\n')
print(f'model accuracy is {score[1]}')
```

```
import matplotlib.pyplot  as plt
plt.plot(history4.history['loss'])
plt.plot(history4.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.grid()
plt.show()
```

```
93/93 [==============================] - 3s 29ms/step - loss: 0.4054 - accuracy:
0.8965


--------------------------------------------------------------------------------
--------------------


model accuracy is 0.8965049386024475
```



[271]:
```
# Initiliazing the sequential model
model4_1 = Sequential()
# Configuring the parameters
```

```
model4_1.add(LSTM(8, input_shape=(timesteps, input_dim),return_sequences= True))
# Adding a dropout layer
model4_1.add(Dropout(0.3))
model4_1.add(BatchNormalization())

model4_1.add(LSTM(16,return_sequences= True))
model4_1.add(Dropout(0.5))
model4_1.add(BatchNormalization())

model4_1.add(LSTM(32))
model4_1.add(Dropout(0.7))
model4_1.add(BatchNormalization())

# Adding a dense output layer with sigmoid activation
model4_1.add(Dense(n_classes, activation='sigmoid'))
model4_1.summary()

# Compiling the model
model4_1.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

```
Model: "sequential_46"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_78 (LSTM)               (None, 128, 8)            576
_________________________________________________________________
dropout_68 (Dropout)         (None, 128, 8)            0
_________________________________________________________________
batch_normalization_39 (Batc (None, 128, 8)            32
_________________________________________________________________
lstm_79 (LSTM)               (None, 128, 16)           1600
_________________________________________________________________
dropout_69 (Dropout)         (None, 128, 16)           0
_________________________________________________________________
batch_normalization_40 (Batc (None, 128, 16)           64
_________________________________________________________________
lstm_80 (LSTM)               (None, 32)                6272
_________________________________________________________________
dropout_70 (Dropout)         (None, 32)                0
_________________________________________________________________
batch_normalization_41 (Batc (None, 32)                128
_________________________________________________________________
dense_32 (Dense)             (None, 6)                 198
=================================================================
Total params: 8,870
```

```
Trainable params: 8,758
Non-trainable params: 112
_________________________________________________________________
```

[272]:

```
model4_1.load_weights('model4.hdf5')

# Compiling the model
model4_1.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

[274]:

```
score = model4_1.evaluate(X_train, Y_train)
print('\n')
print('-'*100)
print('\n')
print(f'model accuracy is {score[1]}')
```

```
230/230 [==============================] - 6s 28ms/step - loss: 0.1911 -
accuracy: 0.9482


--------------------------------------------------------------------------------
--------------------


model accuracy is 0.9481773376464844
```

[277]:

```
# Initiliazing the sequential model
model5 = Sequential()
# Configuring the parameters
model5.add(LSTM(16, input_shape=(timesteps, input_dim),return_sequences= True))
# Adding a dropout layer
model5.add(Dropout(0.3))
model5.add(BatchNormalization())

model5.add(LSTM(32,return_sequences= True))
model5.add(Dropout(0.5))
model5.add(BatchNormalization())

model5.add(LSTM(64))
model5.add(Dropout(0.7))
model5.add(BatchNormalization())

# Adding a dense output layer with sigmoid activation
model5.add(Dense(n_classes, activation='sigmoid'))
model5.summary()

# Compiling the model
```

```
model5.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

```
Model: "sequential_48"
-----------------------------------------------------------------
Layer (type)                 Output Shape              Param #
=================================================================
lstm_84 (LSTM)               (None, 128, 16)           1664
-----------------------------------------------------------------
dropout_74 (Dropout)         (None, 128, 16)           0
-----------------------------------------------------------------
batch_normalization_45 (Batc (None, 128, 16)           64
-----------------------------------------------------------------
lstm_85 (LSTM)               (None, 128, 32)           6272
-----------------------------------------------------------------
dropout_75 (Dropout)         (None, 128, 32)           0
-----------------------------------------------------------------
batch_normalization_46 (Batc (None, 128, 32)           128
-----------------------------------------------------------------
lstm_86 (LSTM)               (None, 64)                24832
-----------------------------------------------------------------
dropout_76 (Dropout)         (None, 64)                0
-----------------------------------------------------------------
batch_normalization_47 (Batc (None, 64)                256
-----------------------------------------------------------------
dense_34 (Dense)             (None, 6)                 390
=================================================================
Total params: 33,606
Trainable params: 33,382
Non-trainable params: 224
-----------------------------------------------------------------
```

[281]:

```
filepath = "model5.hdf5"

# Keep only a single checkpoint, the best over test accuracy.
checkpoint = tf.keras.callbacks.ModelCheckpoint(filepath,
                            monitor='val_accuracy',
                            verbose=1,
                            save_best_only=True,
                            mode='auto')

# Training the model
history5=model5.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_split=0.33,
          epochs=epochs,
          callbacks=[checkpoint])
```

```
Epoch 1/30
308/308 [==============================] - ETA: 0s - loss: 0.3002 - accuracy:
0.8960
Epoch 00001: val_accuracy improved from -inf to 0.92048, saving model to
model5.hdf5
308/308 [==============================] - 55s 178ms/step - loss: 0.3002 -
accuracy: 0.8960 - val_loss: 0.3959 - val_accuracy: 0.9205
Epoch 2/30
308/308 [==============================] - ETA: 0s - loss: 0.2605 - accuracy:
0.9129
Epoch 00002: val_accuracy improved from 0.92048 to 0.92419, saving model to
model5.hdf5
308/308 [==============================] - 53s 171ms/step - loss: 0.2605 -
accuracy: 0.9129 - val_loss: 0.3494 - val_accuracy: 0.9242
Epoch 3/30
308/308 [==============================] - ETA: 0s - loss: 0.2778 - accuracy:
0.9131
Epoch 00003: val_accuracy improved from 0.92419 to 0.92707, saving model to
model5.hdf5
308/308 [==============================] - 52s 170ms/step - loss: 0.2778 -
accuracy: 0.9131 - val_loss: 0.3488 - val_accuracy: 0.9271
Epoch 4/30
308/308 [==============================] - ETA: 0s - loss: 0.2308 - accuracy:
0.9184
Epoch 00004: val_accuracy improved from 0.92707 to 0.92913, saving model to
model5.hdf5
308/308 [==============================] - 52s 169ms/step - loss: 0.2308 -
accuracy: 0.9184 - val_loss: 0.3894 - val_accuracy: 0.9291
Epoch 5/30
308/308 [==============================] - ETA: 0s - loss: 0.2125 - accuracy:
0.9269
Epoch 00005: val_accuracy improved from 0.92913 to 0.93737, saving model to
model5.hdf5
308/308 [==============================] - 53s 172ms/step - loss: 0.2125 -
accuracy: 0.9269 - val_loss: 0.3099 - val_accuracy: 0.9374
Epoch 6/30
308/308 [==============================] - ETA: 0s - loss: 0.2183 - accuracy:
0.9267
Epoch 00006: val_accuracy did not improve from 0.93737
308/308 [==============================] - 52s 168ms/step - loss: 0.2183 -
accuracy: 0.9267 - val_loss: 0.3374 - val_accuracy: 0.9089
Epoch 7/30
308/308 [==============================] - ETA: 0s - loss: 0.2039 - accuracy:
0.9279
```

```
Epoch 00007: val_accuracy did not improve from 0.93737
308/308 [==============================] - 53s 172ms/step - loss: 0.2039 -
accuracy: 0.9279 - val_loss: 0.3103 - val_accuracy: 0.9361
Epoch 8/30
308/308 [==============================] - ETA: 0s - loss: 0.1777 - accuracy:
0.9356
Epoch 00008: val_accuracy did not improve from 0.93737
308/308 [==============================] - 53s 173ms/step - loss: 0.1777 -
accuracy: 0.9356 - val_loss: 0.3726 - val_accuracy: 0.9308
Epoch 9/30
308/308 [==============================] - ETA: 0s - loss: 0.1928 - accuracy:
0.9312
Epoch 00009: val_accuracy did not improve from 0.93737
308/308 [==============================] - 53s 171ms/step - loss: 0.1928 -
accuracy: 0.9312 - val_loss: 0.4570 - val_accuracy: 0.9110
Epoch 10/30
308/308 [==============================] - ETA: 0s - loss: 0.1831 - accuracy:
0.9328
Epoch 00010: val_accuracy did not improve from 0.93737
308/308 [==============================] - 52s 169ms/step - loss: 0.1831 -
accuracy: 0.9328 - val_loss: 0.3272 - val_accuracy: 0.9374
Epoch 11/30
308/308 [==============================] - ETA: 0s - loss: 0.1960 - accuracy:
0.9328
Epoch 00011: val_accuracy did not improve from 0.93737
308/308 [==============================] - 53s 172ms/step - loss: 0.1960 -
accuracy: 0.9328 - val_loss: 0.3646 - val_accuracy: 0.9188
Epoch 12/30
308/308 [==============================] - ETA: 0s - loss: 0.1964 - accuracy:
0.9324
Epoch 00012: val_accuracy did not improve from 0.93737
308/308 [==============================] - 53s 172ms/step - loss: 0.1964 -
accuracy: 0.9324 - val_loss: 0.3689 - val_accuracy: 0.9081
Epoch 13/30
308/308 [==============================] - ETA: 0s - loss: 0.1635 - accuracy:
0.9375
Epoch 00013: val_accuracy did not improve from 0.93737
308/308 [==============================] - 55s 179ms/step - loss: 0.1635 -
accuracy: 0.9375 - val_loss: 0.3493 - val_accuracy: 0.9337
Epoch 14/30
308/308 [==============================] - ETA: 0s - loss: 0.1735 - accuracy:
0.9391
Epoch 00014: val_accuracy did not improve from 0.93737
308/308 [==============================] - 52s 169ms/step - loss: 0.1735 -
accuracy: 0.9391 - val_loss: 0.4975 - val_accuracy: 0.8879
Epoch 15/30
308/308 [==============================] - ETA: 0s - loss: 0.1724 - accuracy:
0.9401
```

```
Epoch 00015: val_accuracy did not improve from 0.93737
308/308 [==============================] - 52s 170ms/step - loss: 0.1724 -
accuracy: 0.9401 - val_loss: 0.3664 - val_accuracy: 0.9135
Epoch 16/30
308/308 [==============================] - ETA: 0s - loss: 0.1810 - accuracy:
0.9352
Epoch 00016: val_accuracy did not improve from 0.93737
308/308 [==============================] - 53s 171ms/step - loss: 0.1810 -
accuracy: 0.9352 - val_loss: 0.4608 - val_accuracy: 0.8929
Epoch 17/30
308/308 [==============================] - ETA: 0s - loss: 0.1708 - accuracy:
0.9391
Epoch 00017: val_accuracy did not improve from 0.93737
308/308 [==============================] - 54s 176ms/step - loss: 0.1708 -
accuracy: 0.9391 - val_loss: 0.3385 - val_accuracy: 0.9180
Epoch 18/30
308/308 [==============================] - ETA: 0s - loss: 0.1692 - accuracy:
0.9360
Epoch 00018: val_accuracy did not improve from 0.93737
308/308 [==============================] - 53s 171ms/step - loss: 0.1692 -
accuracy: 0.9360 - val_loss: 0.5704 - val_accuracy: 0.8912
Epoch 19/30
308/308 [==============================] - ETA: 0s - loss: 0.1749 - accuracy:
0.9366
Epoch 00019: val_accuracy did not improve from 0.93737
308/308 [==============================] - 56s 183ms/step - loss: 0.1749 -
accuracy: 0.9366 - val_loss: 0.4495 - val_accuracy: 0.9238
Epoch 20/30
308/308 [==============================] - ETA: 0s - loss: 0.1872 - accuracy:
0.9332
Epoch 00020: val_accuracy did not improve from 0.93737
308/308 [==============================] - 53s 170ms/step - loss: 0.1872 -
accuracy: 0.9332 - val_loss: 0.3663 - val_accuracy: 0.9324
Epoch 21/30
308/308 [==============================] - ETA: 0s - loss: 0.1565 - accuracy:
0.9415
Epoch 00021: val_accuracy did not improve from 0.93737
308/308 [==============================] - 52s 168ms/step - loss: 0.1565 -
accuracy: 0.9415 - val_loss: 0.3407 - val_accuracy: 0.9345
Epoch 22/30
308/308 [==============================] - ETA: 0s - loss: 0.1439 - accuracy:
0.9421
Epoch 00022: val_accuracy did not improve from 0.93737
308/308 [==============================] - 52s 170ms/step - loss: 0.1439 -
accuracy: 0.9421 - val_loss: 0.2479 - val_accuracy: 0.9250
Epoch 23/30
308/308 [==============================] - ETA: 0s - loss: 0.1564 - accuracy:
0.9403
```

```
Epoch 00023: val_accuracy improved from 0.93737 to 0.93778, saving model to
model5.hdf5
308/308 [==============================] - 53s 171ms/step - loss: 0.1564 -
accuracy: 0.9403 - val_loss: 0.2317 - val_accuracy: 0.9378
Epoch 24/30
308/308 [==============================] - ETA: 0s - loss: 0.1565 - accuracy:
0.9417
Epoch 00024: val_accuracy did not improve from 0.93778
308/308 [==============================] - 52s 169ms/step - loss: 0.1565 -
accuracy: 0.9417 - val_loss: 0.5119 - val_accuracy: 0.9159
Epoch 25/30
308/308 [==============================] - ETA: 0s - loss: 0.1419 - accuracy:
0.9409
Epoch 00025: val_accuracy did not improve from 0.93778
308/308 [==============================] - 58s 187ms/step - loss: 0.1419 -
accuracy: 0.9409 - val_loss: 0.3861 - val_accuracy: 0.9151
Epoch 26/30
308/308 [==============================] - ETA: 0s - loss: 0.1576 - accuracy:
0.9411
Epoch 00026: val_accuracy did not improve from 0.93778
308/308 [==============================] - 53s 171ms/step - loss: 0.1576 -
accuracy: 0.9411 - val_loss: 0.3488 - val_accuracy: 0.9151
Epoch 27/30
308/308 [==============================] - ETA: 0s - loss: 0.1557 - accuracy:
0.9468
Epoch 00027: val_accuracy did not improve from 0.93778
308/308 [==============================] - 52s 169ms/step - loss: 0.1557 -
accuracy: 0.9468 - val_loss: 0.4483 - val_accuracy: 0.9230
Epoch 28/30
308/308 [==============================] - ETA: 0s - loss: 0.1457 - accuracy:
0.9448
Epoch 00028: val_accuracy did not improve from 0.93778
308/308 [==============================] - 53s 171ms/step - loss: 0.1457 -
accuracy: 0.9448 - val_loss: 0.5598 - val_accuracy: 0.9073
Epoch 29/30
308/308 [==============================] - ETA: 0s - loss: 0.1468 - accuracy:
0.9458
Epoch 00029: val_accuracy did not improve from 0.93778
308/308 [==============================] - 52s 169ms/step - loss: 0.1468 -
accuracy: 0.9458 - val_loss: 0.4036 - val_accuracy: 0.9155
Epoch 30/30
308/308 [==============================] - ETA: 0s - loss: 0.1406 - accuracy:
0.9436
Epoch 00030: val_accuracy did not improve from 0.93778
308/308 [==============================] - 52s 169ms/step - loss: 0.1406 -
accuracy: 0.9436 - val_loss: 0.5308 - val_accuracy: 0.8949
```

[282]:

```
# Confusion Matrix
print(confusion_matrix(Y_test, model5.predict(X_test)))
```

```
Pred                LAYING  SITTING  ...  WALKING_DOWNSTAIRS  WALKING_UPSTAIRS
True                                 ...
LAYING                 510        0  ...                   0                 2
SITTING                  1      436  ...                   0                 0
STANDING                 0      151  ...                   0                 0
WALKING                  0        0  ...                  18                 0
WALKING_DOWNSTAIRS       0        0  ...                 417                 2
WALKING_UPSTAIRS         0        0  ...                   4               455

[6 rows x 6 columns]
```

[283]:

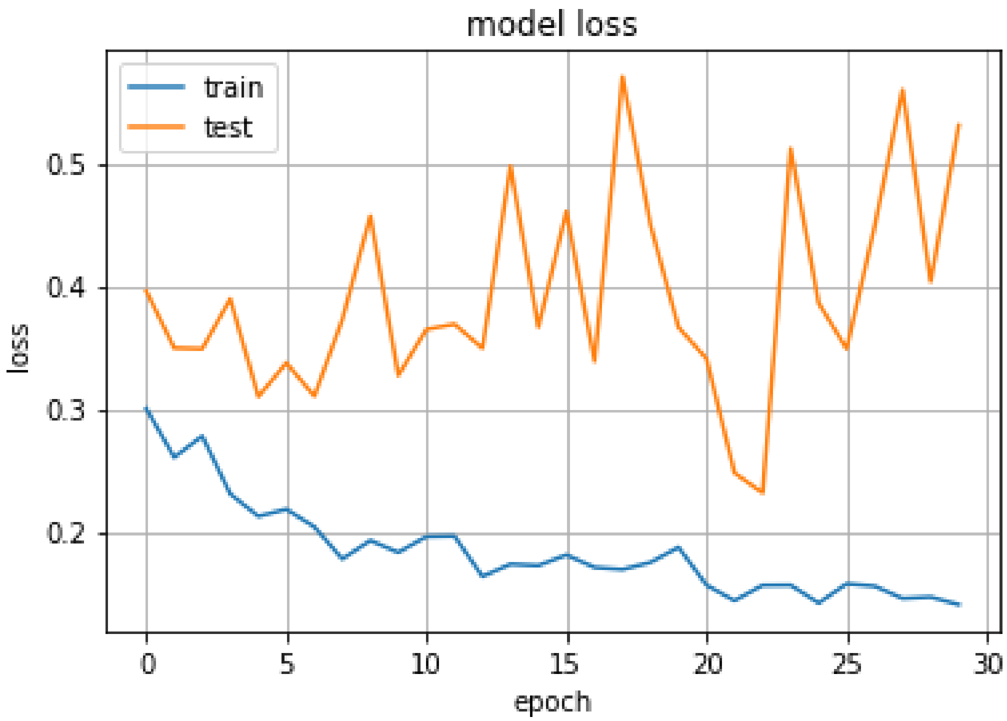
```
score = model5.evaluate(X_test, Y_test)
print('\n')
print('-'*100)
print('\n')
print(f'model accuracy is {score[1]}')

import matplotlib.pyplot  as plt
plt.plot(history5.history['loss'])
plt.plot(history5.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.grid()
plt.show()
```

```
93/93 [==============================] - 3s 37ms/step - loss: 0.3429 - accuracy:
0.9084


--------------------------------------------------------------------------------
--------------------


model accuracy is 0.9083814024925232
```



[285]:

```
# Initiliazing the sequential model
model5_1 = Sequential()
# Configuring the parameters
model5_1.add(LSTM(16, input_shape=(timesteps, input_dim),return_sequences=␣
 ↪True))
# Adding a dropout layer
model5_1.add(Dropout(0.3))
model5_1.add(BatchNormalization())

model5_1.add(LSTM(32,return_sequences= True))
model5_1.add(Dropout(0.5))
model5_1.add(BatchNormalization())

model5_1.add(LSTM(64))
model5_1.add(Dropout(0.7))
model5_1.add(BatchNormalization())

# Adding a dense output layer with sigmoid activation
model5_1.add(Dense(n_classes, activation='sigmoid'))
model5_1.summary()

# Compiling the model
model5_1.load_weights('model5.hdf5')
```

```
# Compiling the model
model5_1.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

```
Model: "sequential_50"
-----------------------------------------------------------------
Layer (type)                 Output Shape              Param #
=================================================================
lstm_90 (LSTM)               (None, 128, 16)           1664
-----------------------------------------------------------------
dropout_80 (Dropout)         (None, 128, 16)           0
-----------------------------------------------------------------
batch_normalization_51 (Batc (None, 128, 16)           64
-----------------------------------------------------------------
lstm_91 (LSTM)               (None, 128, 32)           6272
-----------------------------------------------------------------
dropout_81 (Dropout)         (None, 128, 32)           0
-----------------------------------------------------------------
batch_normalization_52 (Batc (None, 128, 32)           128
-----------------------------------------------------------------
lstm_92 (LSTM)               (None, 64)                24832
-----------------------------------------------------------------
dropout_82 (Dropout)         (None, 64)                0
-----------------------------------------------------------------
batch_normalization_53 (Batc (None, 64)                256
-----------------------------------------------------------------
dense_36 (Dense)             (None, 6)                 390
=================================================================
Total params: 33,606
Trainable params: 33,382
Non-trainable params: 224
-----------------------------------------------------------------
```

[286]:

```
score = model5_1.evaluate(X_test, Y_test)
print('\n')
print('-'*100)
print('\n')
print(f'model accuracy is {score[1]}')
```

```
93/93 [==============================] - 4s 39ms/step - loss: 0.3214 - accuracy:
0.9328


--------------------------------------------------------------------------------
--------------------
```

```
model accuracy is 0.9328130483627319
```

[289]:

```
from prettytable import PrettyTable
x = PrettyTable()
x.field_names = ["S.NO.", "architecture", "Test Accuracy"]
x.add_row(["1", "LSTM(8+16)", "70.00%"])
x.add_row(["2", "LSTM(16+32)","90.91%"])
x.add_row(["3", "LSTM(32+64)","92.23%"])
x.add_row(["4", "LSTM(8+16+32)","94.82%"])
x.add_row(["5", "LSTM(16+32+64)","93.28%"])
print(x)
```

```
+-------+----------------+---------------+
| S.NO. |  architecture  | Test Accuracy |
+-------+----------------+---------------+
|   1   |   LSTM(8+16)   |     70.00%    |
|   2   |  LSTM(16+32)   |     90.91%    |
|   3   |  LSTM(32+64)   |     92.23%    |
|   4   | LSTM(8+16+32)  |     94.82%    |
|   5   | LSTM(16+32+64) |     93.28%    |
+-------+----------------+---------------+
```