In [1]:

```
# Importing Libraries
```

In [2]:

```
import pandas as pd
import numpy as np
```

In [3]:

```
# 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'])
```

Data

In [4]:

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

In [5]:

```
# 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"
]
```

In [6]:

```
# 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).as_matrix()
        )

    # 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))
```

In [7]:

```
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).as_matrix()
```

In [8]:

```
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
```

In [9]:

```
# Importing tensorflow
np.random.seed(42)
import tensorflow as tf
tf.set_random_seed(42)
```

In [10]:

```
# Configuring a session
session_conf = tf.ConfigProto(
    intra_op_parallelism_threads=1,
    inter_op_parallelism_threads=1
)
```

In [11]:

```
# Import Keras
from keras import backend as K
sess = tf.Session(graph=tf.get_default_graph(), config=session_conf)
K.set_session(sess)
```

```
Using TensorFlow backend.
```

In [12]:

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

In [13]:

```
# Initializing parameters
```

```
epochs = 30
batch_size = 16
n_hidden = 32
```

In [14]:

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

In [15]:

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

```
C:\Users\DELL\Anaconda3\lib\site-packages\ipykernel_launcher.py:12: FutureWarning: Method
.as_matrix will be removed in a future version. Use .values instead.
  if sys.path[0] == '':
C:\Users\DELL\Anaconda3\lib\site-packages\ipykernel_launcher.py:11: FutureWarning: Method
.as_matrix will be removed in a future version. Use .values instead.
  # This is added back by InteractiveShellApp.init_path()
```

In [16]:

```
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

In [17]:

```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(n_hidden, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model.add(Dropout(0.5))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

```
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-
packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling
BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint i
s deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
lstm_1 (LSTM)                (None, 32)                5376      
_________________________________________________________________
dropout_1 (Dropout)          (None, 32)                0         
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 198       
=================================================================
Total params: 5,574
Trainable params: 5,574
Non-trainable params: 0
```

```
_________________________________________________________________
```

In [18]:

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

In [23]:

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

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 92s 13ms/step - loss: 1.3018 - acc: 0.4395 - val_loss
: 1.1254 - val_acc: 0.4662
Epoch 2/30
7352/7352 [==============================] - 94s 13ms/step - loss: 0.9666 - acc: 0.5880 - val_loss
: 0.9491 - val_acc: 0.5714
Epoch 3/30
7352/7352 [==============================] - 97s 13ms/step - loss: 0.7812 - acc: 0.6408 - val_loss
: 0.8286 - val_acc: 0.5850
Epoch 4/30
7352/7352 [==============================] - 95s 13ms/step - loss: 0.6941 - acc: 0.6574 - val_loss
: 0.7297 - val_acc: 0.6128
Epoch 5/30
7352/7352 [==============================] - 92s 13ms/step - loss: 0.6336 - acc: 0.6912 - val_loss
: 0.7359 - val_acc: 0.6787
Epoch 6/30
7352/7352 [==============================] - 94s 13ms/step - loss: 0.5859 - acc: 0.7134 - val_loss
: 0.7015 - val_acc: 0.6939
Epoch 7/30
7352/7352 [==============================] - 95s 13ms/step - loss: 0.5692 - acc: 0.7477 - val_loss
: 0.5995 - val_acc: 0.7387
Epoch 8/30
7352/7352 [==============================] - 96s 13ms/step - loss: 0.4899 - acc: 0.7809 - val_loss
: 0.5762 - val_acc: 0.7387
Epoch 9/30
7352/7352 [==============================] - 90s 12ms/step - loss: 0.4482 - acc: 0.7886 - val_loss
: 0.7413 - val_acc: 0.7126
Epoch 10/30
7352/7352 [==============================] - 90s 12ms/step - loss: 0.4132 - acc: 0.8077 - val_loss
: 0.5048 - val_acc: 0.7513
Epoch 11/30
7352/7352 [==============================] - 89s 12ms/step - loss: 0.3985 - acc: 0.8274 - val_loss
: 0.5234 - val_acc: 0.7452
Epoch 12/30
7352/7352 [==============================] - 91s 12ms/step - loss: 0.3378 - acc: 0.8638 - val_loss
: 0.4114 - val_acc: 0.8833
Epoch 13/30
7352/7352 [==============================] - 91s 12ms/step - loss: 0.2947 - acc: 0.9051 - val_loss
: 0.4386 - val_acc: 0.8731
Epoch 14/30
7352/7352 [==============================] - 90s 12ms/step - loss: 0.2448 - acc: 0.9291 - val_loss
: 0.3768 - val_acc: 0.8921
Epoch 15/30
7352/7352 [==============================] - 91s 12ms/step - loss: 0.2157 - acc: 0.9331 - val_loss
: 0.4441 - val_acc: 0.8931
Epoch 16/30
7352/7352 [==============================] - 90s 12ms/step - loss: 0.2053 - acc: 0.9366 - val_loss
: 0.4162 - val_acc: 0.8968
Epoch 17/30
7352/7352 [==============================] - 89s 12ms/step - loss: 0.2028 - acc: 0.9404 - val_loss
: 0.4538 - val_acc: 0.8962
Epoch 18/30
7352/7352 [==============================] - 93s 13ms/step - loss: 0.1911 - acc: 0.9419 - val_loss
: 0.3964 - val_acc: 0.8999
Epoch 19/30
7352/7352 [==============================] - 96s 13ms/step - loss: 0.1912 - acc: 0.9407 - val_loss
```

```
: 0.3165 - val_acc: 0.9030
Epoch 20/30
7352/7352 [==============================] - 96s 13ms/step - loss: 0.1732 - acc: 0.9446 - val_loss
: 0.4546 - val_acc: 0.8904
Epoch 21/30
7352/7352 [==============================] - 94s 13ms/step - loss: 0.1782 - acc: 0.9444 - val_loss
: 0.3346 - val_acc: 0.9063
Epoch 22/30
7352/7352 [==============================] - 95s 13ms/step - loss: 0.1812 - acc: 0.9418 - val_loss
: 0.8164 - val_acc: 0.8582
Epoch 23/30
7352/7352 [==============================] - 95s 13ms/step - loss: 0.1824 - acc: 0.9426 - val_loss
: 0.4240 - val_acc: 0.9036
Epoch 24/30
7352/7352 [==============================] - 94s 13ms/step - loss: 0.1726 - acc: 0.9429 - val_loss
: 0.4067 - val_acc: 0.9148
Epoch 25/30
7352/7352 [==============================] - 96s 13ms/step - loss: 0.1737 - acc: 0.9411 - val_loss
: 0.3396 - val_acc: 0.9074
Epoch 26/30
7352/7352 [==============================] - 96s 13ms/step - loss: 0.1650 - acc: 0.9461 - val_loss
: 0.3806 - val_acc: 0.9019
Epoch 27/30
7352/7352 [==============================] - 89s 12ms/step - loss: 0.1925 - acc: 0.9415 - val_loss
: 0.6464 - val_acc: 0.8850
Epoch 28/30
7352/7352 [==============================] - 91s 12ms/step - loss: 0.1965 - acc: 0.9425 - val_loss
: 0.3363 - val_acc: 0.9203
Epoch 29/30
7352/7352 [==============================] - 92s 12ms/step - loss: 0.1889 - acc: 0.9431 - val_loss
: 0.3737 - val_acc: 0.9158
Epoch 30/30
7352/7352 [==============================] - 95s 13ms/step - loss: 0.1945 - acc: 0.9414 - val_loss
: 0.3088 - val_acc: 0.9097
```

Out[23]:

```
<keras.callbacks.History at 0x29b5ee36a20>
```

In [24]:

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

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 512        0        25        0                   0
SITTING                  3      410        75        0                   0
STANDING                 0       87       445        0                   0
WALKING                  0        0         0      481                   2
WALKING_DOWNSTAIRS       0        0         0        0                 382
WALKING_UPSTAIRS         0        0         0        2                  18

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            3
STANDING                           0
WALKING                           13
WALKING_DOWNSTAIRS                38
WALKING_UPSTAIRS                 451
```

In [27]:

```
score = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 4s 2ms/step
```

In [28]:

```
score
```

Out[28]:

```
[0.3087582236972612, 0.9097387173396675]
```

- With a simple 2 layer architecture we got 90.09% accuracy and a loss of 0.30
- We can further imporve the performace with Hyperparameter tuning

Model -1

In [16]:

```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(40, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model.add(Dropout(0.5))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

```
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-
packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling
BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint i
s deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 40)                8000
_________________________________________________________________
dropout_1 (Dropout)          (None, 40)                0
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 246
=================================================================
Total params: 8,246
Trainable params: 8,246
Non-trainable params: 0
_________________________________________________________________
```

In [17]:

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

In [18]:

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

```
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-
packages\tensorflow_core\python\ops\math_grad.py:1424: where (from
tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:422: The name tf.global_variables is deprecated. Plea
se use tf.compat.v1.global_variables instead.

Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 60s 8ms/step - loss: 1.2751 - accuracy: 0.4566 - val
```

```
loss: 1.1532 - val_accuracy: 0.5168
Epoch 2/30
7352/7352 [==============================] - 59s 8ms/step - loss: 1.0828 - accuracy: 0.5094 - val_
loss: 1.0826 - val_accuracy: 0.4954
Epoch 3/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.9695 - accuracy: 0.5461 - val_
loss: 0.9446 - val_accuracy: 0.5389
Epoch 4/30
7352/7352 [==============================] - 85s 12ms/step - loss: 0.8720 - accuracy: 0.5928 - val
_loss: 1.0721 - val_accuracy: 0.5443
Epoch 5/30
7352/7352 [==============================] - 99s 14ms/step - loss: 0.8074 - accuracy: 0.6288 - val
_loss: 0.7764 - val_accuracy: 0.6115
Epoch 6/30
7352/7352 [==============================] - 103s 14ms/step - loss: 0.6825 - accuracy: 0.6727 - va
l_loss: 0.7068 - val_accuracy: 0.6719
Epoch 7/30
7352/7352 [==============================] - 101s 14ms/step - loss: 0.6128 - accuracy: 0.7402 - va
l_loss: 0.6498 - val_accuracy: 0.7655
Epoch 8/30
7352/7352 [==============================] - 109s 15ms/step - loss: 0.5230 - accuracy: 0.8093 - va
l_loss: 0.6808 - val_accuracy: 0.7720
Epoch 9/30
7352/7352 [==============================] - 114s 15ms/step - loss: 0.4131 - accuracy: 0.8675 - va
l_loss: 0.5582 - val_accuracy: 0.8324
Epoch 10/30
7352/7352 [==============================] - 112s 15ms/step - loss: 0.3565 - accuracy: 0.8887 - va
l_loss: 0.6027 - val_accuracy: 0.8337
Epoch 11/30
7352/7352 [==============================] - 124s 17ms/step - loss: 0.2972 - accuracy: 0.9075 - va
l_loss: 0.7928 - val_accuracy: 0.7631
Epoch 12/30
7352/7352 [==============================] - 132s 18ms/step - loss: 0.2553 - accuracy: 0.9202 - va
l_loss: 0.5271 - val_accuracy: 0.8680
Epoch 13/30
7352/7352 [==============================] - 137s 19ms/step - loss: 0.2389 - accuracy: 0.9263 - va
l_loss: 0.6066 - val_accuracy: 0.8493
Epoch 14/30
7352/7352 [==============================] - 141s 19ms/step - loss: 0.2194 - accuracy: 0.9305 - va
l_loss: 0.4856 - val_accuracy: 0.8853
Epoch 15/30
7352/7352 [==============================] - 151s 21ms/step - loss: 0.2009 - accuracy: 0.9355 - va
l_loss: 0.5948 - val_accuracy: 0.8649
Epoch 16/30
7352/7352 [==============================] - 159s 22ms/step - loss: 0.1887 - accuracy: 0.9343 - va
l_loss: 0.5596 - val_accuracy: 0.8612
Epoch 17/30
7352/7352 [==============================] - 165s 22ms/step - loss: 0.2124 - accuracy: 0.9327 - va
l_loss: 1.1237 - val_accuracy: 0.8015
Epoch 18/30
7352/7352 [==============================] - 174s 24ms/step - loss: 0.2092 - accuracy: 0.9297 - va
l_loss: 0.4727 - val_accuracy: 0.8945
Epoch 19/30
7352/7352 [==============================] - 180s 24ms/step - loss: 0.1829 - accuracy: 0.9393 - va
l_loss: 0.4309 - val_accuracy: 0.8951
Epoch 20/30
7352/7352 [==============================] - 185s 25ms/step - loss: 0.1893 - accuracy: 0.9402 - va
l_loss: 0.4604 - val_accuracy: 0.8985
Epoch 21/30
7352/7352 [==============================] - 191s 26ms/step - loss: 0.1718 - accuracy: 0.9411 - va
l_loss: 0.3586 - val_accuracy: 0.9006
Epoch 22/30
7352/7352 [==============================] - 205s 28ms/step - loss: 0.1796 - accuracy: 0.9430 - va
l_loss: 0.3259 - val_accuracy: 0.9030
Epoch 23/30
7352/7352 [==============================] - 211s 29ms/step - loss: 0.1673 - accuracy: 0.9392 - va
l_loss: 0.2938 - val_accuracy: 0.9158
Epoch 24/30
7352/7352 [==============================] - 219s 30ms/step - loss: 0.1735 - accuracy: 0.9389 - va
l_loss: 0.2817 - val_accuracy: 0.9121
Epoch 25/30
7352/7352 [==============================] - 221s 30ms/step - loss: 0.1603 - accuracy: 0.9448 - va
l_loss: 0.3182 - val_accuracy: 0.9067
Epoch 26/30
7352/7352 [==============================] - 229s 31ms/step - loss: 0.1794 - accuracy: 0.9436 - va
l_loss: 0.2851 - val_accuracy: 0.9155
Epoch 27/30
Epoch 27/30
7352/7352 [==============================] - 242s 33ms/step - loss: 0.1573 - accuracy: 0.9430 - va
l_loss: 0.2766 - val_accuracy: 0.9138
Epoch 28/30
7352/7352 [==============================] - 248s 34ms/step - loss: 0.1474 - accuracy: 0.9480 - va
l_loss: 0.3196 - val_accuracy: 0.9138
Epoch 29/30
7352/7352 [==============================] - 256s 35ms/step - loss: 0.1748 - accuracy: 0.9427 - va
l_loss: 0.3264 - val_accuracy: 0.9199
Epoch 30/30
7352/7352 [==============================] - 267s 36ms/step - loss: 0.1764 - accuracy: 0.9437 - va
l_loss: 0.3167 - val_accuracy: 0.9118
```

Out[18]:

```
<keras.callbacks.callbacks.History at 0x1e8b83a4d08>
```

In [19]:

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

```
Pred                    LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                     537        0         0        0                   0
SITTING                      1      394        93        0                   2
STANDING                     0       87       444        1                   0
WALKING                      0        0         0      443                   2
WALKING_DOWNSTAIRS           0        0         0        1                 414
WALKING_UPSTAIRS             0        0         0        4                  12

Pred                    WALKING_UPSTAIRS
True
LAYING                                 0
SITTING                                1
STANDING                               0
WALKING                               51
WALKING_DOWNSTAIRS                     5
WALKING_UPSTAIRS                     455
```

In [20]:

```
score = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - ETA:  - 10s 4ms/step
```

In [21]:

```
score
```

Out[21]:

```
[0.3166825441067603, 0.9117746949195862]
```

Mode-2

In [17]:

```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(70, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model.add(Dropout(0.7))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

```
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-
```

```
packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling
BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint i
s deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
WARNING:tensorflow:Large dropout rate: 0.7 (>0.5). In TensorFlow 2.x, dropout() uses dropout rate
instead of keep_prob. Please ensure that this is intended.
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 70)                22400
_________________________________________________________________
dropout_1 (Dropout)          (None, 70)                0
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 426
=================================================================
Total params: 22,826
Trainable params: 22,826
Non-trainable params: 0
_________________________________________________________________
```

In [18]:

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

In [24]:

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

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 83s 11ms/step - loss: 1.3200 - accuracy: 0.4222 - val
_loss: 1.1266 - val_accuracy: 0.5056
Epoch 2/30
7352/7352 [==============================] - 82s 11ms/step - loss: 1.0199 - accuracy: 0.5495 - val
_loss: 0.9827 - val_accuracy: 0.5928
Epoch 3/30
7352/7352 [==============================] - 94s 13ms/step - loss: 0.8651 - accuracy: 0.6107 - val
_loss: 0.8543 - val_accuracy: 0.6237
Epoch 4/30
7352/7352 [==============================] - 112s 15ms/step - loss: 0.7704 - accuracy: 0.6425 - va
l_loss: 1.0827 - val_accuracy: 0.6067
Epoch 5/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.7125 - accuracy: 0.6732 - va
l_loss: 0.6864 - val_accuracy: 0.6868
Epoch 6/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.6546 - accuracy: 0.6986 - va
l_loss: 0.7473 - val_accuracy: 0.6444
Epoch 7/30
7352/7352 [==============================] - 149s 20ms/step - loss: 0.5986 - accuracy: 0.7666 - va
l_loss: 0.6916 - val_accuracy: 0.7289
Epoch 8/30
7352/7352 [==============================] - 155s 21ms/step - loss: 0.5637 - accuracy: 0.8085 - va
l_loss: 0.5564 - val_accuracy: 0.7889
Epoch 9/30
7352/7352 [==============================] - 194s 26ms/step - loss: 0.4016 - accuracy: 0.8683 - va
l_loss: 0.4682 - val_accuracy: 0.8381
Epoch 10/30
7352/7352 [==============================] - 207s 28ms/step - loss: 0.3462 - accuracy: 0.8938 - va
l_loss: 0.7120 - val_accuracy: 0.8069
Epoch 11/30
7352/7352 [==============================] - 217s 30ms/step - loss: 0.2898 - accuracy: 0.9128 - va
l_loss: 0.5276 - val_accuracy: 0.8575
Epoch 12/30
7352/7352 [==============================] - 214s 29ms/step - loss: 0.2392 - accuracy: 0.9218 - va
l_loss: 0.3729 - val_accuracy: 0.8880
```

```
Epoch 13/30
7352/7352 [==============================] - 240s 33ms/step - loss: 0.2519 - accuracy: 0.9259 - va
l_loss: 0.4410 - val_accuracy: 0.8856
Epoch 14/30
7352/7352 [==============================] - 247s 34ms/step - loss: 0.2090 - accuracy: 0.9323 - va
l_loss: 1.2566 - val_accuracy: 0.7139
Epoch 15/30
7352/7352 [==============================] - 229s 31ms/step - loss: 0.2240 - accuracy: 0.9309 - va
l_loss: 0.4624 - val_accuracy: 0.8758
Epoch 16/30
7352/7352 [==============================] - 234s 32ms/step - loss: 0.1949 - accuracy: 0.9385 - va
l_loss: 0.3560 - val_accuracy: 0.8996
Epoch 17/30
7352/7352 [==============================] - 238s 32ms/step - loss: 0.1915 - accuracy: 0.9335 - va
l_loss: 0.3072 - val_accuracy: 0.8955
Epoch 18/30
7352/7352 [==============================] - 248s 34ms/step - loss: 0.2030 - accuracy: 0.9387 - va
l_loss: 0.3829 - val_accuracy: 0.9077
Epoch 19/30
7352/7352 [==============================] - 257s 35ms/step - loss: 0.1961 - accuracy: 0.9373 - va
l_loss: 0.3599 - val_accuracy: 0.9050
Epoch 20/30
7352/7352 [==============================] - 270s 37ms/step - loss: 0.1825 - accuracy: 0.9397 - va
l_loss: 0.4031 - val_accuracy: 0.8867
Epoch 21/30
7352/7352 [==============================] - 302s 41ms/step - loss: 0.1630 - accuracy: 0.9411 - va
l_loss: 0.5637 - val_accuracy: 0.8860
Epoch 22/30
7352/7352 [==============================] - 316s 43ms/step - loss: 0.1745 - accuracy: 0.9431 - va
l_loss: 0.8422 - val_accuracy: 0.8697
Epoch 23/30
7352/7352 [==============================] - 332s 45ms/step - loss: 0.1755 - accuracy: 0.9429 - va
l_loss: 0.6011 - val_accuracy: 0.8890
Epoch 24/30
7352/7352 [==============================] - 354s 48ms/step - loss: 0.1615 - accuracy: 0.9464 - va
l_loss: 0.7342 - val_accuracy: 0.8700
Epoch 25/30
7352/7352 [==============================] - 372s 51ms/step - loss: 0.1537 - accuracy: 0.9463 - va
l_loss: 0.5534 - val_accuracy: 0.8918
Epoch 26/30
7352/7352 [==============================] - 353s 48ms/step - loss: 0.1517 - accuracy: 0.9467 - va
l_loss: 0.5197 - val_accuracy: 0.9053
Epoch 27/30
7352/7352 [==============================] - 369s 50ms/step - loss: 0.1727 - accuracy: 0.9448 - va
l_loss: 0.4946 - val_accuracy: 0.8799
Epoch 28/30
7352/7352 [==============================] - 398s 54ms/step - loss: 0.1536 - accuracy: 0.9471 - va
l_loss: 0.3513 - val_accuracy: 0.9087
Epoch 29/30
7352/7352 [==============================] - 458s 62ms/step - loss: nan - accuracy: 0.9091 - val_l
oss: nan - val_accuracy: 0.1683
Epoch 30/30
7352/7352 [==============================] - 451s 61ms/step - loss: nan - accuracy: 0.1668 - val_l
oss: nan - val_accuracy: 0.1683
```

Out[24]:

```
<keras.callbacks.callbacks.History at 0x272ee970308>
```

In [25]:

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

```
Pred                WALKING
True
LAYING                  537
SITTING                 491
STANDING                532
WALKING                 496
WALKING_DOWNSTAIRS      420
WALKING_UPSTAIRS        471
```

In [26]:

In [26]:

```
score = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 10s 3ms/step
```

In [27]:

```
score
```

Out[27]:

```
[nan, 0.16830675303936005]
```

Model-3

In [17]:

```
# code from https://keras.io/regularizers/
from keras.regularizers import L1L2
from keras.models import load_model
from keras.callbacks import ModelCheckpoint
from keras.layers import LSTM , BatchNormalization
reg = L1L2(0.01, 0.01)
```

In [18]:

```
model = Sequential()
model.add(LSTM(100, input_shape=(timesteps, input_dim), kernel_initializer='glorot_normal' , return
_sequences=True, bias_regularizer=reg))
model.add(BatchNormalization())
model.add(Dropout(0.80))
model.add(LSTM(50))
model.add(Dropout(0.80))
model.add(Dense(n_classes, activation='sigmoid'))
print("Model Summary: ")
model.summary()
```

```
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-
packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling
BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint i
s deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
WARNING:tensorflow:Large dropout rate: 0.8 (>0.5). In TensorFlow 2.x, dropout() uses dropout rate
instead of keep_prob. Please ensure that this is intended.
WARNING:tensorflow:Large dropout rate: 0.8 (>0.5). In TensorFlow 2.x, dropout() uses dropout rate
instead of keep_prob. Please ensure that this is intended.
Model Summary:
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 128, 100)          44000
_________________________________________________________________
batch_normalization_1 (Batch (None, 128, 100)          400
_________________________________________________________________
dropout_1 (Dropout)          (None, 128, 100)          0
_________________________________________________________________
lstm_2 (LSTM)                (None, 50)                30200
_________________________________________________________________
dropout_2 (Dropout)          (None, 50)                0
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 306
=================================================================
Total params: 74,906
Trainable params: 74,706
Non-trainable params: 200
_________________________________________________________________
```

In [19]:

```
# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
checkpoint_3 = ModelCheckpoint("model_6.h5",monitor="val_acc",mode="max",save_best_only =
True,verbose=1)
```

In [20]:

```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=15,callbacks=[checkpoint_3])
```

```
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-
packages\tensorflow_core\python\ops\math_grad.py:1424: where (from
tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:422: The name tf.global_variables is deprecated. Plea
se use tf.compat.v1.global_variables instead.

Train on 7352 samples, validate on 2947 samples
Epoch 1/15
7352/7352 [==============================] - 139s 19ms/step - loss: 2.6173 - accuracy: 0.5220 - va
l_loss: 2.5817 - val_accuracy: 0.5029
Epoch 2/15
```

```
C:\Users\DELL\Anaconda3\lib\site-packages\keras\callbacks\callbacks.py:707: RuntimeWarning: Can sa
ve best model only with val_acc available, skipping.
  'skipping.' % (self.monitor), RuntimeWarning)
```

```
7352/7352 [==============================] - 145s 20ms/step - loss: 1.3352 - accuracy: 0.6249 - va
l_loss: 0.8323 - val_accuracy: 0.6288
Epoch 3/15
7352/7352 [==============================] - 154s 21ms/step - loss: 0.8115 - accuracy: 0.6602 - va
l_loss: 0.7513 - val_accuracy: 0.6865
Epoch 4/15
7352/7352 [==============================] - 174s 24ms/step - loss: 0.7281 - accuracy: 0.7172 - va
l_loss: 0.7608 - val_accuracy: 0.7170
Epoch 5/15
7352/7352 [==============================] - 206s 28ms/step - loss: 0.6322 - accuracy: 0.7599 - va
l_loss: 0.5619 - val_accuracy: 0.8062
Epoch 6/15
7352/7352 [==============================] - 228s 31ms/step - loss: 0.5795 - accuracy: 0.7807 - va
l_loss: 0.5914 - val_accuracy: 0.8242
Epoch 7/15
7352/7352 [==============================] - 262s 36ms/step - loss: 0.5288 - accuracy: 0.8217 - va
l_loss: 0.7612 - val_accuracy: 0.8208
Epoch 8/15
7352/7352 [==============================] - 293s 40ms/step - loss: 0.4881 - accuracy: 0.8415 - va
l_loss: 0.4252 - val_accuracy: 0.8704
Epoch 9/15
7352/7352 [==============================] - 322s 44ms/step - loss: 0.4282 - accuracy: 0.8728 - va
l_loss: 0.4716 - val_accuracy: 0.8409
Epoch 10/15
7352/7352 [==============================] - 351s 48ms/step - loss: 0.3854 - accuracy: 0.8834 - va
l_loss: 0.4493 - val_accuracy: 0.8789
Epoch 11/15
7352/7352 [==============================] - 386s 52ms/step - loss: 0.3593 - accuracy: 0.8958 - va
l_loss: 0.4021 - val_accuracy: 0.9145
Epoch 12/15
7352/7352 [==============================] - 415s 56ms/step - loss: nan - accuracy: 0.2928 - val_l
oss: nan - val_accuracy: 0.1683
Epoch 13/15
7352/7352 [==============================] - 445s 60ms/step - loss: nan - accuracy: 0.1668 - val_l
oss: nan - val_accuracy: 0.1683
Epoch 14/15
7352/7352 [==============================] - 509s 69ms/step - loss: nan - accuracy: 0.1668 - val_l
oss: nan - val_accuracy: 0.1683
```

```
oss: nan - val_accuracy: 0.1683
Epoch 15/15
7352/7352 [==============================] - 508s 69ms/step - loss: nan - accuracy: 0.1668 - val_l
oss: nan - val_accuracy: 0.1683
```

Out[20]:

```
<keras.callbacks.callbacks.History at 0x1b5d719a208>
```

In [21]:

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

```
Pred                 WALKING
True
LAYING                   537
SITTING                  491
STANDING                 532
WALKING                  496
WALKING_DOWNSTAIRS       420
WALKING_UPSTAIRS         471
```

In [22]:

```
score = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 50s 17ms/step
```

In [23]:

```
score
```

Out[23]:

```
[nan, 0.16830675303936005]
```

Model-4

In [17]:

```
# code from https://keras.io/regularizers/
from keras.regularizers import L1L2
from keras.models import load_model
from keras.callbacks import ModelCheckpoint
from keras.layers import LSTM , BatchNormalization
reg = L1L2(0.01, 0.01)
```

In [18]:

```
reg = L1L2(0.01, 0.01)
model = Sequential()
model.add(LSTM(100, input_shape=(timesteps, input_dim), kernel_initializer='glorot_normal' , return
_sequences=True, bias_regularizer=reg))
model.add(BatchNormalization())
model.add(Dropout(0.70))
model.add(LSTM(50))
model.add(Dropout(0.70))
model.add(Dense(n_classes, activation='sigmoid'))
print("Model Summary: ")
model.summary()
```

```
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-
packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling
BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint i
s deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
```

```
WARNING:tensorflow:Large dropout rate: 0.7 (>0.5). In TensorFlow 2.x, dropout() uses dropout rate
instead of keep_prob. Please ensure that this is intended.
WARNING:tensorflow:Large dropout rate: 0.7 (>0.5). In TensorFlow 2.x, dropout() uses dropout rate
instead of keep_prob. Please ensure that this is intended.
Model Summary:
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
lstm_1 (LSTM)                (None, 128, 100)          44000     
_________________________________________________________________
batch_normalization_1 (Batch (None, 128, 100)          400       
_________________________________________________________________
dropout_1 (Dropout)          (None, 128, 100)          0         
_________________________________________________________________
lstm_2 (LSTM)                (None, 50)                30200     
_________________________________________________________________
dropout_2 (Dropout)          (None, 50)                0         
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 306       
=================================================================
Total params: 74,906
Trainable params: 74,706
Non-trainable params: 200
_________________________________________________________________
```

In [19]:

```
# Compiling the model
model.compile(loss='binary_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
checkpoint_3 = ModelCheckpoint("model_7.h5",monitor="val_accuracy",mode="max",save_best_only =
True,verbose=1)
```

```
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-
packages\tensorflow_core\python\ops\nn_impl.py:183: where (from tensorflow.python.ops.array_ops) i
s deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
```

In [20]:

```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=15,callbacks=[checkpoint_3])
```

```
WARNING:tensorflow:From C:\Users\DELL\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:422: The name tf.global_variables is deprecated. Plea
se use tf.compat.v1.global_variables instead.

Train on 7352 samples, validate on 2947 samples
Epoch 1/15
7352/7352 [==============================] - 128s 17ms/step - loss: 1.7121 - accuracy: 0.8507 - va
l_loss: 1.1402 - val_accuracy: 0.8754

Epoch 00001: val_accuracy improved from -inf to 0.87535, saving model to model_7.h5
Epoch 2/15
7352/7352 [==============================] - 127s 17ms/step - loss: 0.6552 - accuracy: 0.9049 - va
l_loss: 0.2718 - val_accuracy: 0.9331

Epoch 00002: val_accuracy improved from 0.87535 to 0.93310, saving model to model_7.h5
Epoch 3/15
7352/7352 [==============================] - 127s 17ms/step - loss: 0.1781 - accuracy: 0.9366 - va
l_loss: 0.2116 - val_accuracy: 0.9161

Epoch 00003: val_accuracy did not improve from 0.93310
Epoch 4/15
7352/7352 [==============================] - 130s 18ms/step - loss: 0.1273 - accuracy: 0.9563 - va
l_loss: 0.0980 - val_accuracy: 0.9612
```

```
Epoch 00004: val_accuracy improved from 0.93310 to 0.96120, saving model to model_7.h5
Epoch 5/15
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1041 - accuracy: 0.9651 - va
l_loss: 0.0775 - val_accuracy: 0.9730

Epoch 00005: val_accuracy improved from 0.96120 to 0.97297, saving model to model_7.h5
Epoch 6/15
7352/7352 [==============================] - 135s 18ms/step - loss: 0.0984 - accuracy: 0.9663 - va
l_loss: 0.1148 - val_accuracy: 0.9654

Epoch 00006: val_accuracy did not improve from 0.97297
Epoch 7/15
7352/7352 [==============================] - 135s 18ms/step - loss: 0.0911 - accuracy: 0.9686 - va
l_loss: 0.0731 - val_accuracy: 0.9720

Epoch 00007: val_accuracy did not improve from 0.97297
Epoch 8/15
7352/7352 [==============================] - 136s 18ms/step - loss: 0.0859 - accuracy: 0.9710 - va
l_loss: 0.0944 - val_accuracy: 0.9716

Epoch 00008: val_accuracy did not improve from 0.97297
Epoch 9/15
7352/7352 [==============================] - 137s 19ms/step - loss: 0.0826 - accuracy: 0.9709 - va
l_loss: 0.0995 - val_accuracy: 0.9680

Epoch 00009: val_accuracy did not improve from 0.97297
Epoch 10/15
7352/7352 [==============================] - 139s 19ms/step - loss: 0.0787 - accuracy: 0.9743 - va
l_loss: 0.0786 - val_accuracy: 0.9751

Epoch 00010: val_accuracy improved from 0.97297 to 0.97512, saving model to model_7.h5
Epoch 11/15
7352/7352 [==============================] - 140s 19ms/step - loss: 0.0749 - accuracy: 0.9740 - va
l_loss: 0.0840 - val_accuracy: 0.9751

Epoch 00011: val_accuracy did not improve from 0.97512
Epoch 12/15
7352/7352 [==============================] - 139s 19ms/step - loss: 0.0753 - accuracy: 0.9736 - va
l_loss: 0.1383 - val_accuracy: 0.9669

Epoch 00012: val_accuracy did not improve from 0.97512
Epoch 13/15
7352/7352 [==============================] - 141s 19ms/step - loss: 0.0737 - accuracy: 0.9742 - va
l_loss: 0.1028 - val_accuracy: 0.9712

Epoch 00013: val_accuracy did not improve from 0.97512
Epoch 14/15
7352/7352 [==============================] - 143s 19ms/step - loss: 0.0708 - accuracy: 0.9752 - va
l_loss: 0.1461 - val_accuracy: 0.9615

Epoch 00014: val_accuracy did not improve from 0.97512
Epoch 15/15
7352/7352 [==============================] - 142s 19ms/step - loss: 0.0691 - accuracy: 0.9752 - va
l_loss: 0.1638 - val_accuracy: 0.9656

Epoch 00015: val_accuracy did not improve from 0.97512
```

Out[20]:

```
<keras.callbacks.callbacks.History at 0x26251c261c8>
```

In [21]:

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

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 537        0         0        0                   0
SITTING                  0      362       128        0                   0
STANDING                 0       47       485        0                   0
WALKING                  0        1         0      462                  33
WALKING_DOWNSTAIRS       0        0         0        0                 420
WALKING_UPSTAIRS         0        0         6       53                  33
```

```
Pred                 WALKING_UPSTAIRS
True
LAYING                              0
SITTING                             1
STANDING                            0
WALKING                             0
WALKING_DOWNSTAIRS                  0
WALKING_UPSTAIRS                  379
```

In [22]:

```
score = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 7s 2ms/step
```

In [23]:

```
score
```

Out[23]:

```
[0.16383741364825719, 0.9655581712722778]
```

In [19]:

```
from prettytable import PrettyTable
x = PrettyTable()
x.field_names = ["Model","Description", "Test loss", "Test Accuracy"]
x.add_row(["1","1 Layer of LSTM(40)","0.3166", "0.9117"])
x.add_row(["2","1 Layer of LSTM(70)+Dropout(0.7)","nan", "0.1683"])
x.add_row(["3","2 Layer of LSTM","nan", "0.1683"])
x.add_row(["4","2 layers of LSTM, BN, Binarycross_entropy","0.1638", "0.9655"])
print(x)
```

```
+-------+-------------------------------------------+-----------+---------------+
| Model |                Description                | Test loss | Test Accuracy |
+-------+-------------------------------------------+-----------+---------------+
|   1   |            1 Layer of LSTM(40)            |   0.3166  |     0.9117    |
|   2   |      1 Layer of LSTM(70)+Dropout(0.7)     |    nan    |     0.1683    |
|   3   |              2 Layer of LSTM              |    nan    |     0.1683    |
|   4   | 2 layers of LSTM, BN, Binarycross_entropy |   0.1638  |     0.9655    |
+-------+-------------------------------------------+-----------+---------------+
```

In [ ]:

```
```