In [1]:

```
# Importing Libraries
```

HumanActivityRecognition

This project is to build a model that predicts the human activities such as Walking, Walking_Upstairs, Walking_Downstairs, Sitting, Standing or Laying.

This dataset is collected from 30 persons(referred as subjects in this dataset), performing different activities with a smartphone to their waists. The data is recorded with the help of sensors (accelerometer and Gyroscope) in that smartphone. This experiment was video recorded to label the data manually.

How data was recorded

By using the sensors(Gyroscope and accelerometer) in a smartphone, they have captured '3-axial linear acceleration'(*tAcc-XYZ*) from accelerometer and '3-axial angular velocity' (*tGyro-XYZ*) from Gyroscope with several variations.

Train and test data were saperated

- The readings from ***70%*** of the volunteers were taken as ***trianing data*** and remaining ***30%*** subjects recordings were taken for ***test data***

Data Size :

> 27 MB

Quick overview of the dataset :

- Accelerometer and Gyroscope readings are taken from 30 volunteers(referred as subjects) while performing the following 6 Activities.

    1. Walking
    2. WalkingUpstairs
    3. WalkingDownstairs
    4. Standing
    5. Sitting
    6. Lying.

- Readings are divided into a window of 2.56 seconds with 50% overlapping.
- Accelerometer readings are divided into gravity acceleration and body acceleration readings, which has x,y and z components each.
- Gyroscope readings are the measure of angular velocities which has x,y and z components.
- Jerk signals are calculated for BodyAcceleration readings.
- Fourier Transforms are made on the above time readings to obtain frequency readings.
- Now, on all the base signal readings., mean, max, mad, sma, arcoefficient, engerybands,entropy etc., are calculated for each window.
- We get a feature vector of 561 features and these features are given in the dataset.
- Each window of readings is a datapoint of 561 features.

Problem Framework

- 30 subjects(volunteers) data is randomly split to 70%(21) test and 30%(7) train data.
- Each datapoint corresponds one of the 6 Activities.

Problem Statement

- Given a new datapoint we have to predict the Activity

How data set is working for deep learning

- Here the time series data is divided in to several windows
- At that window consider what is the accrelometer X,Y,Z axis signals and what is gyroscope X,Y,Z signals and what task is performed by the invidule at the current window
- Considering these information several windows are created with the respective output class
- take all the windows for all the 21 person (as discussed 70% of data train data) as train data
- take all the windows for data of 7 person as test

Our plan to improve the accuracy

- The Low RAM machine may take some time to train the model, so we are going to use less data for hyperparameter tuning.
- Some of the training data set can be used to Cross validate the model.
- When we will finish all our hyper parameter tuning we can use all data to train teh model and use test data set to get the accuracy of the model

In [1]:

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

In [2]:

```
# 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 [3]:

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

In [4]:

```
# 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 [5]:

```
# 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 [6]:

```
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 [7]:

```
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 [8]:

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

In [9]:

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

In [10]:

```
# 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 [11]:

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

In [24]:

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

#First we will use 32 hidden layers and then we will try to increase the layer to see the impact on acc
uracy
n_hidden = 32
```

In [12]:

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

In [13]:

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

```
C:\Users\bolua\AppData\Local\Continuum\anaconda3\lib\site-packages\ipykernel_launcher.py:12: FutureWarn
ing: Method .as_matrix will be removed in a future version. Use .values instead.
  if sys.path[0] == '':
```

Dividing data for CV

In [15]:

```
#Get dlesser data for hyper parameter tuning
X_train_forHP = X_train[:1000]
X_CV_forHP = X_train[1000:1500]
Y_train_forHP = Y_train[:1000]
Y_CV_forHP = Y_train[1000:1500]
```

In [17]:

```
timesteps = len(X_train_forHP[0])
input_dim = len(X_train_forHP[0][0])
n_classes = _count_classes(Y_train_forHP)

print(timesteps)
print(input_dim)
print(len(X_train_forHP))
```

```
128
9
1000
```

one LSTM layer - 32 hidden layer - 50% dropout

In [18]:

```
# 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\bolua\AppData\Local\Continuum\anaconda3\lib\site-packages\tensorflow\p
ython\framework\op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecat
ed and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From C:\Users\bolua\AppData\Local\Continuum\anaconda3\lib\site-packages\keras\backen
d\tensorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is dep
recated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
_________________________________________________________________
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 [19]:

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

In [20]:

```
# Training the model
model.fit(X_train_forHP,
          Y_train_forHP,
          batch_size=batch_size,
          validation_data=(X_CV_forHP, Y_CV_forHP),
          epochs=epochs)
```

```
WARNING:tensorflow:From C:\Users\bolua\AppData\Local\Continuum\anaconda3\lib\site-packages\tensorflow\p
ython\ops\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and will be re
moved in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 1000 samples, validate on 500 samples
Epoch 1/30
1000/1000 [==============================] - 6s 6ms/step - loss: 1.6213 - acc: 0.3540 - val_loss: 1.455
5 - val_acc: 0.4700
Epoch 2/30
1000/1000 [==============================] - 4s 4ms/step - loss: 1.4084 - acc: 0.4250 - val_loss: 1.475
9 - val_acc: 0.3940
Epoch 3/30
1000/1000 [==============================] - 4s 4ms/step - loss: 1.3150 - acc: 0.4610 - val_loss: 1.387
3 - val_acc: 0.4080
Epoch 4/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.2834 - acc: 0.4720 - val_loss: 1.200
7 - val_acc: 0.4660
Epoch 5/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.2583 - acc: 0.4870 - val_loss: 1.213
5 - val_acc: 0.4520
```

```
Epoch 6/30
1000/1000 [==============================] - 4s 4ms/step - loss: 1.2064 - acc: 0.4700 - val_loss: 1.173
8 - val_acc: 0.4520
Epoch 7/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.1617 - acc: 0.4790 - val_loss: 1.363
6 - val_acc: 0.3860
Epoch 8/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.1401 - acc: 0.4970 - val_loss: 1.251
3 - val_acc: 0.4220
Epoch 9/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.1422 - acc: 0.4820 - val_loss: 1.326
3 - val_acc: 0.3560
Epoch 10/30
1000/1000 [==============================] - 4s 4ms/step - loss: 1.1133 - acc: 0.4920 - val_loss: 1.123
0 - val_acc: 0.5080
Epoch 11/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.1086 - acc: 0.4940 - val_loss: 1.183
1 - val_acc: 0.4120
Epoch 12/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.0983 - acc: 0.5090 - val_loss: 1.117
3 - val_acc: 0.4740
Epoch 13/30
1000/1000 [==============================] - 4s 4ms/step - loss: 1.0961 - acc: 0.4790 - val_loss: 1.188
0 - val_acc: 0.4020
Epoch 14/30
1000/1000 [==============================] - 4s 4ms/step - loss: 1.0848 - acc: 0.4990 - val_loss: 1.130
0 - val_acc: 0.4460
Epoch 15/30
1000/1000 [==============================] - 4s 4ms/step - loss: 1.0505 - acc: 0.5100 - val_loss: 1.045
9 - val_acc: 0.5200
Epoch 16/30
1000/1000 [==============================] - 4s 4ms/step - loss: 1.0388 - acc: 0.4970 - val_loss: 0.965
2 - val_acc: 0.5120
Epoch 17/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.9895 - acc: 0.5300 - val_loss: 1.122
7 - val_acc: 0.3980
Epoch 18/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.9599 - acc: 0.5610 - val_loss: 1.040
5 - val_acc: 0.5000
Epoch 19/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.9291 - acc: 0.5830 - val_loss: 1.065
6 - val_acc: 0.4800
Epoch 20/30
1000/1000 [==============================] - 4s 4ms/step - loss: 0.8839 - acc: 0.5970 - val_loss: 0.955
1 - val_acc: 0.5460
Epoch 21/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.8807 - acc: 0.6140 - val_loss: 0.852
4 - val_acc: 0.5520
Epoch 22/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.8029 - acc: 0.6410 - val_loss: 0.908
7 - val_acc: 0.5280
Epoch 23/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7657 - acc: 0.6190 - val_loss: 0.836
1 - val_acc: 0.5360
Epoch 24/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7676 - acc: 0.6290 - val_loss: 0.837
4 - val_acc: 0.5420
Epoch 25/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7748 - acc: 0.6230 - val_loss: 0.758
0 - val_acc: 0.5680
Epoch 26/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7331 - acc: 0.6440 - val_loss: 0.814
3 - val_acc: 0.5720
Epoch 27/30
1000/1000 [==============================] - 4s 4ms/step - loss: 0.7208 - acc: 0.6620 - val_loss: 0.697
1 - val_acc: 0.6380
Epoch 28/30
1000/1000 [==============================] - 4s 4ms/step - loss: 0.7175 - acc: 0.6580 - val_loss: 0.850
1 - val_acc: 0.5740
Epoch 29/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7004 - acc: 0.6500 - val_loss: 0.836
0 - val_acc: 0.6100
Epoch 30/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7071 - acc: 0.6520 - val_loss: 0.696
9 - val_acc: 0.6000
```

Out[20]:

Out[20]:

```
<keras.callbacks.History at 0x215385d7160>
```

In [21]:

```
score = model.evaluate(X_CV_forHP, Y_CV_forHP)
score
```

```
500/500 [==============================] - 0s 549us/step
```

Out[21]:

```
[0.6969475907087326, 0.6]
```

In [44]:

```
from prettytable import PrettyTable

out_table = PrettyTable()
#x.del_row(1)
out_table.field_names = ["MODEL", "Hidden unit","Dropout","Accurcy"]
```

In [45]:

```
out_table.add_row(["LSTM One","32","50%",score])
print(out_table)
```

```
+----------+-------------+---------+---------------------------+
|  MODEL   | Hidden unit | Dropout |          Accurcy          |
+----------+-------------+---------+---------------------------+
| LSTM One |      32     |   50%   | [0.6969475907087326, 0.6] |
+----------+-------------+---------+---------------------------+
```

one LSTM layer - 64 hidden layer - 50% dropout

In [24]:

```
n_hidden = 64
```

In [25]:

```
# 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()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_2 (LSTM)                (None, 64)                18944
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 390
=================================================================
Total params: 19,334
Trainable params: 19,334
Non-trainable params: 0
_________________________________________________________________
```

In [26]:

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

In [27]:

```
# Training the model
model.fit(X_train_forHP,
          Y_train_forHP,
          batch_size=batch_size,
          validation_data=(X_CV_forHP, Y_CV_forHP),
          epochs=epochs)
```

```
Train on 1000 samples, validate on 500 samples
Epoch 1/30
1000/1000 [==============================] - 6s 6ms/step - loss: 1.5488 - acc: 0.3290 - val_loss: 1.402
0 - val_acc: 0.3400
Epoch 2/30
1000/1000 [==============================] - 6s 6ms/step - loss: 1.3493 - acc: 0.3590 - val_loss: 1.406
3 - val_acc: 0.3400
Epoch 3/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.3410 - acc: 0.3700 - val_loss: 1.350
4 - val_acc: 0.3400
Epoch 4/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.2700 - acc: 0.3910 - val_loss: 1.383
8 - val_acc: 0.3400
Epoch 5/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.2269 - acc: 0.4200 - val_loss: 1.239
2 - val_acc: 0.3940
Epoch 6/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.1589 - acc: 0.4880 - val_loss: 1.165
4 - val_acc: 0.4140
Epoch 7/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.0848 - acc: 0.5280 - val_loss: 1.388
5 - val_acc: 0.3820
Epoch 8/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.0982 - acc: 0.4840 - val_loss: 1.159
6 - val_acc: 0.5100
Epoch 9/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.0907 - acc: 0.5160 - val_loss: 1.000
2 - val_acc: 0.5200
Epoch 10/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.9482 - acc: 0.5530 - val_loss: 0.952
5 - val_acc: 0.5300
Epoch 11/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.9780 - acc: 0.5730 - val_loss: 1.071
0 - val_acc: 0.5440
Epoch 12/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.8526 - acc: 0.5790 - val_loss: 0.842
1 - val_acc: 0.5860
Epoch 13/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.8407 - acc: 0.6150 - val_loss: 1.014
0 - val_acc: 0.5060
Epoch 14/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.8248 - acc: 0.6040 - val_loss: 0.816
4 - val_acc: 0.5620
Epoch 15/30
1000/1000 [==============================] - 6s 6ms/step - loss: 0.7545 - acc: 0.6390 - val_loss: 0.851
0 - val_acc: 0.5240
Epoch 16/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7525 - acc: 0.6340 - val_loss: 1.090
0 - val_acc: 0.5900
Epoch 17/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7191 - acc: 0.6320 - val_loss: 0.715
2 - val_acc: 0.6380
Epoch 18/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7093 - acc: 0.6330 - val_loss: 0.750
1 - val_acc: 0.5640
Epoch 19/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6993 - acc: 0.6430 - val_loss: 0.760
4 - val_acc: 0.5720
Epoch 20/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6761 - acc: 0.6470 - val_loss: 0.847
7 - val_acc: 0.5660
```

```
Epoch 21/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6662 - acc: 0.6500 - val_loss: 0.924
4 - val_acc: 0.5900
Epoch 22/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6649 - acc: 0.6480 - val_loss: 0.791
5 - val_acc: 0.6360
Epoch 23/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6865 - acc: 0.6530 - val_loss: 0.686
5 - val_acc: 0.6040
Epoch 24/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6566 - acc: 0.6550 - val_loss: 0.699
8 - val_acc: 0.5880
Epoch 25/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6545 - acc: 0.6520 - val_loss: 0.796
0 - val_acc: 0.6100
Epoch 26/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6225 - acc: 0.6550 - val_loss: 0.624
6 - val_acc: 0.6400
Epoch 27/30
1000/1000 [==============================] - 6s 6ms/step - loss: 0.5628 - acc: 0.6760 - val_loss: 0.641
9 - val_acc: 0.6240
Epoch 28/30
1000/1000 [==============================] - 6s 6ms/step - loss: 0.5862 - acc: 0.6720 - val_loss: 0.531
2 - val_acc: 0.6420
Epoch 29/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6319 - acc: 0.6600 - val_loss: 0.553
8 - val_acc: 0.6320
Epoch 30/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6611 - acc: 0.6840 - val_loss: 1.081
8 - val_acc: 0.5660
```

Out[27]:

```
<keras.callbacks.History at 0x21542d04828>
```

In [28]:

```
score = model.evaluate(X_CV_forHP, Y_CV_forHP)
score
```

```
500/500 [==============================] - 1s 1ms/step
```

Out[28]:

```
[1.0818304223418236, 0.566]
```

In [46]:

```
out_table.add_row(["LSTM One","64","50%",score])
print(out_table)
```

```
+----------+-------------+---------+-----------------------------+
|  MODEL   | Hidden unit | Dropout |           Accurcy           |
+----------+-------------+---------+-----------------------------+
| LSTM One |      32     |   50%   |  [0.6969475907087326, 0.6]  |
| LSTM One |      64     |   50%   | [1.0818304223418236, 0.566] |
+----------+-------------+---------+-----------------------------+
```

one LSTM layer - 64 hidden layer - 70% dropout

In [30]:

```
# 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()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_3 (LSTM)                (None, 64)                18944
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 6)                 390
=================================================================
Total params: 19,334
Trainable params: 19,334
Non-trainable params: 0
_________________________________________________________________
```

In [31]:

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

In [32]:

```
# Training the model
model.fit(X_train_forHP,
          Y_train_forHP,
          batch_size=batch_size,
          validation_data=(X_CV_forHP, Y_CV_forHP),
          epochs=epochs)
```

```
Train on 1000 samples, validate on 500 samples
Epoch 1/30
1000/1000 [==============================] - 6s 6ms/step - loss: 1.5288 - acc: 0.3420 - val_loss: 1.439
5 - val_acc: 0.3400
Epoch 2/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.3274 - acc: 0.3790 - val_loss: 1.215
2 - val_acc: 0.4760
Epoch 3/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.2601 - acc: 0.4330 - val_loss: 1.224
9 - val_acc: 0.4180
Epoch 4/30
1000/1000 [==============================] - 6s 6ms/step - loss: 1.1956 - acc: 0.4630 - val_loss: 1.093
5 - val_acc: 0.4980
Epoch 5/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.1587 - acc: 0.4830 - val_loss: 1.101
3 - val_acc: 0.4900
Epoch 6/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.0997 - acc: 0.4950 - val_loss: 1.092
5 - val_acc: 0.4700
Epoch 7/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.0996 - acc: 0.4810 - val_loss: 1.129
9 - val_acc: 0.4960
Epoch 8/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.0523 - acc: 0.5110 - val_loss: 1.141
0 - val_acc: 0.4620
Epoch 9/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.1104 - acc: 0.4860 - val_loss: 1.092
9 - val_acc: 0.4600
Epoch 10/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.1038 - acc: 0.4960 - val_loss: 1.375
6 - val_acc: 0.3400
Epoch 11/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.0121 - acc: 0.5250 - val_loss: 0.935
1 - val_acc: 0.4800
Epoch 12/30
1000/1000 [==============================] - 5s 5ms/step - loss: 1.0272 - acc: 0.5100 - val_loss: 1.277
7 - val_acc: 0.3700
Epoch 13/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.9554 - acc: 0.5410 - val_loss: 0.935
5 - val_acc: 0.5160
Epoch 14/30
```

```
1000/1000 [==============================] - 5s 5ms/step - loss: 0.8960 - acc: 0.5890 - val_loss: 0.901
9 - val_acc: 0.5620
Epoch 15/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.8100 - acc: 0.6370 - val_loss: 0.970
4 - val_acc: 0.5320
Epoch 16/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7768 - acc: 0.6420 - val_loss: 1.133
0 - val_acc: 0.5100
Epoch 17/30
1000/1000 [==============================] - 6s 6ms/step - loss: 0.7351 - acc: 0.6530 - val_loss: 1.258
0 - val_acc: 0.5220
Epoch 18/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7285 - acc: 0.6310 - val_loss: 1.106
1 - val_acc: 0.5240
Epoch 19/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.7357 - acc: 0.6690 - val_loss: 0.840
4 - val_acc: 0.5920
Epoch 20/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6716 - acc: 0.6800 - val_loss: 1.054
3 - val_acc: 0.5660
Epoch 21/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.5817 - acc: 0.7170 - val_loss: 0.851
0 - val_acc: 0.6600
Epoch 22/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.5191 - acc: 0.7420 - val_loss: 0.979
2 - val_acc: 0.5640
Epoch 23/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.5984 - acc: 0.7410 - val_loss: 0.945
3 - val_acc: 0.6640
Epoch 24/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.6928 - acc: 0.7000 - val_loss: 0.980
5 - val_acc: 0.6800
Epoch 25/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.5472 - acc: 0.7470 - val_loss: 1.052
5 - val_acc: 0.6040
Epoch 26/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.5208 - acc: 0.7890 - val_loss: 0.879
7 - val_acc: 0.7820
Epoch 27/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.4201 - acc: 0.8210 - val_loss: 0.624
8 - val_acc: 0.7280
Epoch 28/30
1000/1000 [==============================] - 5s 5ms/step - loss: 0.4044 - acc: 0.8270 - val_loss: 0.626
8 - val_acc: 0.7420
Epoch 29/30
1000/1000 [==============================] - 6s 6ms/step - loss: 0.3473 - acc: 0.8600 - val_loss: 0.516
0 - val_acc: 0.7320
Epoch 30/30
1000/1000 [==============================] - 6s 6ms/step - loss: 0.3909 - acc: 0.8470 - val_loss: 0.914
9 - val_acc: 0.8040
```

Out[32]:

```
<keras.callbacks.History at 0x215449022e8>
```

In [33]:

```
score = model.evaluate(X_CV_forHP, Y_CV_forHP)
score
```

```
500/500 [==============================] - 0s 564us/step
```

Out[33]:

```
[0.9148820471763611, 0.804]
```

In [47]:

```
out_table.add_row(["LSTM One","64","70%",score])
print(out_table)
```

```
+----------+-------------+---------+----------------------------+
|  MODEL   | Hidden unit | Dropout |           Accurcy          |
```

```
+----------+-------------+---------+-----------------------------+
| LSTM One |      32     |   50%   |  [0.6969475907087326, 0.6]  |
| LSTM One |      64     |   50%   | [1.0818304223418236, 0.566] |
| LSTM One |      64     |   70%   | [0.9148820471763611, 0.804] |
+----------+-------------+---------+-----------------------------+
```

Two LSTM layer - 64 hidden layer - 80% dropout

In [39]:

```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(n_hidden, input_shape=(timesteps, input_dim), return_sequences=True))
# Adding a dropout layer
model.add(Dropout(0.8))
# Configuring the parameters for second layer
model.add(LSTM(n_hidden))
# Adding a dropout layer
model.add(Dropout(0.8))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_12 (LSTM)               (None, 128, 64)           18944
_________________________________________________________________
dropout_8 (Dropout)          (None, 128, 64)           0
_________________________________________________________________
lstm_13 (LSTM)               (None, 64)                33024
_________________________________________________________________
dropout_9 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_4 (Dense)              (None, 6)                 390
=================================================================
Total params: 52,358
Trainable params: 52,358
Non-trainable params: 0
_________________________________________________________________
```

In [40]:

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

In [41]:

```
# Training the model
model.fit(X_train_forHP,
          Y_train_forHP,
          batch_size=batch_size,
          validation_data=(X_CV_forHP, Y_CV_forHP),
          epochs=epochs)
```

```
Train on 1000 samples, validate on 500 samples
Epoch 1/30
1000/1000 [==============================] - 13s 13ms/step - loss: 1.5306 - acc: 0.3520 - val_loss: 1.4
723 - val_acc: 0.3540
Epoch 2/30
1000/1000 [==============================] - 12s 12ms/step - loss: 1.3313 - acc: 0.4190 - val_loss: 1.4
218 - val_acc: 0.3700
Epoch 3/30
1000/1000 [==============================] - 11s 11ms/step - loss: 1.2558 - acc: 0.4750 - val_loss: 1.1
182 - val_acc: 0.5240
Epoch 4/30
1000/1000 [==============================] - 11s 11ms/step - loss: 1.1926 - acc: 0.4590 - val_loss: 0.9
996 - val_acc: 0.5000
```

```
Epoch 5/30
1000/1000 [==============================] - 11s 11ms/step - loss: 1.1339 - acc: 0.4790 - val_loss: 0.9
673 - val_acc: 0.5840
Epoch 6/30
1000/1000 [==============================] - 11s 11ms/step - loss: 1.0977 - acc: 0.5360 - val_loss: 1.0
050 - val_acc: 0.5180
Epoch 7/30
1000/1000 [==============================] - 11s 11ms/step - loss: 1.0290 - acc: 0.5600 - val_loss: 0.8
968 - val_acc: 0.4880
Epoch 8/30
1000/1000 [==============================] - 12s 12ms/step - loss: 0.9896 - acc: 0.5540 - val_loss: 0.8
172 - val_acc: 0.6100
Epoch 9/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.9323 - acc: 0.5660 - val_loss: 0.9
400 - val_acc: 0.4800
Epoch 10/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.8649 - acc: 0.5920 - val_loss: 0.7
374 - val_acc: 0.6200
Epoch 11/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.8125 - acc: 0.6380 - val_loss: 0.7
088 - val_acc: 0.6500
Epoch 12/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.8798 - acc: 0.6030 - val_loss: 0.6
902 - val_acc: 0.6380
Epoch 13/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.8636 - acc: 0.6310 - val_loss: 0.6
823 - val_acc: 0.6200
Epoch 14/30
1000/1000 [==============================] - 12s 12ms/step - loss: 0.8082 - acc: 0.6260 - val_loss: 0.7
373 - val_acc: 0.6220
Epoch 15/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.7606 - acc: 0.6340 - val_loss: 0.8
136 - val_acc: 0.6040
Epoch 16/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.9245 - acc: 0.6270 - val_loss: 0.9
955 - val_acc: 0.5360
Epoch 17/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.7777 - acc: 0.6390 - val_loss: 0.7
320 - val_acc: 0.6080
Epoch 18/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.7806 - acc: 0.6420 - val_loss: 0.8
958 - val_acc: 0.5340
Epoch 19/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.7409 - acc: 0.6450 - val_loss: 0.9
034 - val_acc: 0.6040
Epoch 20/30
1000/1000 [==============================] - 12s 12ms/step - loss: 0.7228 - acc: 0.6620 - val_loss: 0.9
161 - val_acc: 0.5980
Epoch 21/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.7713 - acc: 0.6640 - val_loss: 0.6
461 - val_acc: 0.6160
Epoch 22/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.7254 - acc: 0.6570 - val_loss: 0.7
471 - val_acc: 0.6080
Epoch 23/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.7450 - acc: 0.6540 - val_loss: 0.8
421 - val_acc: 0.5820
Epoch 24/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.7770 - acc: 0.6540 - val_loss: 0.9
913 - val_acc: 0.5880
Epoch 25/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.6905 - acc: 0.6550 - val_loss: 1.1
264 - val_acc: 0.5660
Epoch 26/30
1000/1000 [==============================] - 12s 12ms/step - loss: 0.6816 - acc: 0.6770 - val_loss: 0.8
076 - val_acc: 0.5660
Epoch 27/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.6922 - acc: 0.6820 - val_loss: 1.3
070 - val_acc: 0.5540
Epoch 28/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.7043 - acc: 0.6650 - val_loss: 0.8
709 - val_acc: 0.5960
Epoch 29/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.6400 - acc: 0.6700 - val_loss: 1.0
458 - val_acc: 0.5820
Epoch 30/30
1000/1000 [==============================] - 11s 11ms/step - loss: 0.7086 - acc: 0.6590 - val_loss: 1.1
```

```
1000/1000 [==============================] - 11s 11ms/step - loss: 0.7000 - acc: 0.6390 - val_loss: 1.1
372 - val_acc: 0.5780
```

Out[41]:

```
<keras.callbacks.History at 0x21547c938d0>
```

In [42]:

```
score = model.evaluate(X_CV_forHP, Y_CV_forHP)
score
```

```
500/500 [==============================] - 1s 1ms/step
```

Out[42]:

```
[1.1371510267443954, 0.578]
```

In [48]:

```
out_table.add_row(["LSTM Two","64","80%",score])
print(out_table)
```

```
+----------+-------------+---------+------------------------------+
|  MODEL   | Hidden unit | Dropout |           Accurcy            |
+----------+-------------+---------+------------------------------+
| LSTM One |      32     |   50%   |  [0.6969475907087326, 0.6]   |
| LSTM One |      64     |   50%   | [1.0818304223418236, 0.566] |
| LSTM One |      64     |   70%   | [0.9148820471763611, 0.804] |
| LSTM Two |      64     |   80%   | [1.1371510267443954, 0.578] |
+----------+-------------+---------+------------------------------+
```

The best performance is given by single layer but high drop out, we will applye the same to train out model

Model training after hyerparameter tuning

In [18]:

```
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 [45]:

```
# 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(0.8))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
```

```
=================================================================
lstm_14 (LSTM)               (None, 64)                18944
_________________________________________________________________
dropout_10 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_5 (Dense)              (None, 6)                 390
=================================================================
Total params: 19,334
Trainable params: 19,334
Non-trainable params: 0
_________________________________________________________________
```

In [46]:

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

In [47]:

```
# 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 [==============================] - 40s 5ms/step - loss: 1.3636 - acc: 0.4369 - val_loss: 1.32
74 - val_acc: 0.4788
Epoch 2/30
7352/7352 [==============================] - 37s 5ms/step - loss: 1.1235 - acc: 0.5180 - val_loss: 0.97
95 - val_acc: 0.6298
Epoch 3/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.9340 - acc: 0.6055 - val_loss: 0.83
66 - val_acc: 0.6281
Epoch 4/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.7991 - acc: 0.6413 - val_loss: 0.80
78 - val_acc: 0.5931
Epoch 5/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.7527 - acc: 0.6609 - val_loss: 0.74
04 - val_acc: 0.6376
Epoch 6/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.7243 - acc: 0.6737 - val_loss: 0.84
92 - val_acc: 0.6166
Epoch 7/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.6700 - acc: 0.7146 - val_loss: 0.82
28 - val_acc: 0.6597
Epoch 8/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.5954 - acc: 0.7485 - val_loss: 0.64
11 - val_acc: 0.7598
Epoch 9/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.5669 - acc: 0.7938 - val_loss: 0.54
49 - val_acc: 0.8161
Epoch 10/30
7352/7352 [==============================] - 40s 5ms/step - loss: 0.4741 - acc: 0.8354 - val_loss: 0.52
33 - val_acc: 0.8402
Epoch 11/30
7352/7352 [==============================] - 38s 5ms/step - loss: 0.4002 - acc: 0.8833 - val_loss: 0.36
74 - val_acc: 0.8605
Epoch 12/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.3855 - acc: 0.8923 - val_loss: 0.35
42 - val_acc: 0.8853
Epoch 13/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.3284 - acc: 0.9052 - val_loss: 0.47
40 - val_acc: 0.8697
Epoch 14/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.2896 - acc: 0.9240 - val_loss: 0.29
79 - val_acc: 0.8958
Epoch 15/30
7352/7352 [==============================] - 38s 5ms/step - loss: 0.2931 - acc: 0.9197 - val_loss: 0.50
27 - val_acc: 0.8941
Epoch 16/30
```

```
Epoch 16/30
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.3742 - val_loss: nan - v
al_acc: 0.1683
Epoch 17/30
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 18/30
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 19/30
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 20/30
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 21/30
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 22/30
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 23/30
7352/7352 [==============================] - 36s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 24/30
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 25/30
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 26/30
7352/7352 [==============================] - 36s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 27/30
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 28/30
7352/7352 [==============================] - 36s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 29/30
7352/7352 [==============================] - 38s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 30/30
7352/7352 [==============================] - 39s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
```

Out[47]:

```
<keras.callbacks.History at 0x2154953df28>
```

In [48]:

```
# 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 [49]:

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

```
2947/2947 [==============================] - 2s 701us/step
```

In [50]:

```
score
```

```
score
```

Out[50]:

```
[nan, 0.168306752629793]
```

In [51]:

```
out_table.add_row(["LSTM One - Highdata","64","80%",score])
print(out_table)
```

```
+--------------------+-------------+---------+-----------------------------+
|       MODEL        | Hidden unit | Dropout |           Accurcy           |
+--------------------+-------------+---------+-----------------------------+
|      LSTM One      |      32     |   50%   |  [0.6969475907087326, 0.6]  |
|      LSTM One      |      64     |   50%   | [1.0818304223418236, 0.566] |
|      LSTM One      |      64     |   70%   | [0.9148820471763611, 0.804] |
|      LSTM Two      |      64     |   80%   | [1.1371510267443954, 0.578] |
| LSTM One - Highdata |      64     |   80%   |   [nan, 0.168306752629793]  |
+--------------------+-------------+---------+-----------------------------+
```

In the above calculation we are hitting the problem of overfillting. To deal with it we will try to reduce the number of epochs and dropout rate and we will observe the performance

We will use relu function to observe the performance

2nd model

In [52]:

```
# 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(0.7))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='relu'))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_15 (LSTM)               (None, 64)                18944
_________________________________________________________________
dropout_11 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_6 (Dense)              (None, 6)                 390
=================================================================
Total params: 19,334
Trainable params: 19,334
Non-trainable params: 0
_________________________________________________________________
```

In [53]:

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

In [54]:

```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=20)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/20
7352/7352 [==============================] - 40s 5ms/step - loss: 2.4433 - acc: 0.4192 - val_loss: 1.46
34 - val_acc: 0.5321
Epoch 2/20
7352/7352 [==============================] - 39s 5ms/step - loss: 1.2398 - acc: 0.5926 - val_loss: nan
- val_acc: 0.6057
Epoch 3/20
7352/7352 [==============================] - 37s 5ms/step - loss: 1.2727 - acc: 0.6238 - val_loss: 2.40
88 - val_acc: 0.5538
Epoch 4/20
7352/7352 [==============================] - 38s 5ms/step - loss: 1.0173 - acc: 0.6485 - val_loss: 1.07
36 - val_acc: 0.6712
Epoch 5/20
7352/7352 [==============================] - 37s 5ms/step - loss: 1.1089 - acc: 0.6778 - val_loss: 1.36
73 - val_acc: 0.6535
Epoch 6/20
7352/7352 [==============================] - 38s 5ms/step - loss: nan - acc: 0.3011 - val_loss: nan - v
al_acc: 0.1683
Epoch 7/20
7352/7352 [==============================] - 38s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 8/20
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 9/20
7352/7352 [==============================] - 38s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 10/20
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 11/20
7352/7352 [==============================] - 38s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 12/20
7352/7352 [==============================] - 38s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 13/20
7352/7352 [==============================] - 38s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 14/20
7352/7352 [==============================] - 39s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 15/20
7352/7352 [==============================] - 37s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 16/20
7352/7352 [==============================] - 38s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 17/20
7352/7352 [==============================] - 38s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 18/20
7352/7352 [==============================] - 38s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 19/20
7352/7352 [==============================] - 38s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
Epoch 20/20
7352/7352 [==============================] - 35s 5ms/step - loss: nan - acc: 0.1668 - val_loss: nan - v
al_acc: 0.1683
```

Out[54]:

```
<keras.callbacks.History at 0x2154e0a5828>
```

In [55]:

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

```
2947/2947 [==============================] - 2s 633us/step
```

Out[55]:

```
[nan, 0.168306752629793]
```

In [52]:

```
out_table.add_row(["LSTM One - Highdata + Relu","64","70%",score])
print(out_table)
```

```
+----------------------------+-------------+---------+----------------------------+
|           MODEL            | Hidden unit | Dropout |           Accurcy          |
+----------------------------+-------------+---------+----------------------------+
|          LSTM One          |      32     |   50%   |  [0.6969475907087326, 0.6] |
|          LSTM One          |      64     |   50%   | [1.0818304223418236, 0.566] |
|          LSTM One          |      64     |   70%   | [0.9148820471763611, 0.804] |
|          LSTM Two          |      64     |   80%   | [1.1371510267443954, 0.578] |
|    LSTM One - Highdata     |      64     |   80%   |   [nan, 0.168306752629793] |
| LSTM One - Highdata + Relu |      64     |   70%   |   [nan, 0.168306752629793] |
+----------------------------+-------------+---------+----------------------------+
```

We got again the problem of overfillting. To deal with it we will try to reduce the number of epochs and dropout rate and we will observe the performance

We will use not use relu function, insted we will go with sigmoid and observe the performance

3rd model

In [57]:

```
# 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(0.7))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_16 (LSTM)               (None, 64)                18944
_________________________________________________________________
dropout_12 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_7 (Dense)              (None, 6)                 390
=================================================================
Total params: 19,334
Trainable params: 19,334
Non-trainable params: 0
_________________________________________________________________
```

In [58]:

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

In [59]:

```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=20)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/20
7352/7352 [==============================] - 38s 5ms/step - loss: 1.2937 - acc: 0.4396 - val_loss: 1.18
04 - val_acc: 0.4425
Epoch 2/20
7352/7352 [==============================] - 36s 5ms/step - loss: 1.1719 - acc: 0.4819 - val_loss: 1.13
33 - val_acc: 0.5154
Epoch 3/20
7352/7352 [==============================] - 42s 6ms/step - loss: 0.9666 - acc: 0.5766 - val_loss: 0.97
99 - val_acc: 0.6142
Epoch 4/20
7352/7352 [==============================] - 41s 6ms/step - loss: 0.8432 - acc: 0.6383 - val_loss: 0.91
14 - val_acc: 0.5993
Epoch 5/20
7352/7352 [==============================] - 40s 5ms/step - loss: 0.6987 - acc: 0.7031 - val_loss: 0.83
02 - val_acc: 0.7133
Epoch 6/20
7352/7352 [==============================] - 41s 6ms/step - loss: 0.6278 - acc: 0.7384 - val_loss: 1.45
43 - val_acc: 0.5823
Epoch 7/20
7352/7352 [==============================] - 42s 6ms/step - loss: 0.6403 - acc: 0.7456 - val_loss: 0.68
44 - val_acc: 0.7648
Epoch 8/20
7352/7352 [==============================] - 41s 6ms/step - loss: 0.5694 - acc: 0.7665 - val_loss: 0.61
38 - val_acc: 0.8113
Epoch 9/20
7352/7352 [==============================] - 42s 6ms/step - loss: 0.4911 - acc: 0.8313 - val_loss: 0.65
17 - val_acc: 0.8039
Epoch 10/20
7352/7352 [==============================] - 43s 6ms/step - loss: 0.4256 - acc: 0.8592 - val_loss: 0.50
80 - val_acc: 0.8168
Epoch 11/20
7352/7352 [==============================] - 42s 6ms/step - loss: 0.4291 - acc: 0.8555 - val_loss: 0.37
64 - val_acc: 0.8724
Epoch 12/20
7352/7352 [==============================] - 43s 6ms/step - loss: 0.3852 - acc: 0.8690 - val_loss: 0.77
10 - val_acc: 0.8107
Epoch 13/20
7352/7352 [==============================] - 41s 6ms/step - loss: 0.3426 - acc: 0.8916 - val_loss: 0.53
61 - val_acc: 0.8395
Epoch 14/20
7352/7352 [==============================] - 43s 6ms/step - loss: 0.3066 - acc: 0.9066 - val_loss: 0.39
73 - val_acc: 0.8697
Epoch 15/20
7352/7352 [==============================] - 42s 6ms/step - loss: 0.2435 - acc: 0.9259 - val_loss: 0.39
25 - val_acc: 0.8836
Epoch 16/20
7352/7352 [==============================] - 41s 6ms/step - loss: 0.2519 - acc: 0.9208 - val_loss: 0.35
85 - val_acc: 0.8914
Epoch 17/20
7352/7352 [==============================] - 43s 6ms/step - loss: 0.2532 - acc: 0.9255 - val_loss: 0.51
06 - val_acc: 0.8778
Epoch 18/20
7352/7352 [==============================] - 43s 6ms/step - loss: 0.2420 - acc: 0.9253 - val_loss: 0.34
91 - val_acc: 0.8958
Epoch 19/20
7352/7352 [==============================] - 42s 6ms/step - loss: 0.2509 - acc: 0.9267 - val_loss: 0.49
10 - val_acc: 0.8975
Epoch 20/20
7352/7352 [==============================] - 43s 6ms/step - loss: 0.2569 - acc: 0.9260 - val_loss: 0.43
61 - val_acc: 0.9023
```

Out[59]:

```
<keras.callbacks.History at 0x21550c8bda0>
```

In [60]:

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

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 508        0         0        0                   0
```

```
SITTING                  0      416        70         2                  0
STANDING                 0      123       406         2                  0
WALKING                  0        0         0       482                  6
WALKING_DOWNSTAIRS       0        0         0         2                415
WALKING_UPSTAIRS         0        1         0        34                  4
```

```
Pred                WALKING_UPSTAIRS
True
LAYING                            29
SITTING                            3
STANDING                           1
WALKING                            8
WALKING_DOWNSTAIRS                 3
WALKING_UPSTAIRS                 432
```

In [61]:

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

```
2947/2947 [==============================] - 2s 700us/step
```

Out[61]:

```
[0.43612632637826976, 0.9022734984730234]
```

In [53]:

```
out_table.add_row(["LSTM One - Highdata + sigmoid","64","70%",score])
print(out_table)
```

```
+------------------------------+-------------+---------+-----------------------------------------+
|              MODEL           | Hidden unit | Dropout |                 Accurcy                 |
+------------------------------+-------------+---------+-----------------------------------------+
|            LSTM One          |      32     |   50%   |         [0.6969475907087326, 0.6]       |
|            LSTM One          |      64     |   50%   |        [1.0818304223418236, 0.566]      |
|            LSTM One          |      64     |   70%   |        [0.9148820471763611, 0.804]      |
|            LSTM Two          |      64     |   80%   |        [1.1371510267443954, 0.578]      |
|      LSTM One - Highdata     |      64     |   80%   |          [nan, 0.168306752629793]       |
|   LSTM One - Highdata + Relu |      64     |   70%   |          [nan, 0.168306752629793]       |
| LSTM One - Highdata + sigmoid |      64     |   70%   | [0.43612632637826976, 0.9022734984730234] |
+------------------------------+-------------+---------+-----------------------------------------+
```

Lets Judge the performance by increasing the number of epochs

4th model

In [65]:

```
# 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(0.7))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_17 (LSTM)               (None, 64)                18944
_________________________________________________________________
dropout_13 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_8 (Dense)              (None, 6)                 390
=================================================================
Total params: 19,334
```

```
Trainable params: 19,334
Non-trainable params: 0
_________________________________________________________________
```

In [66]:

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

In [67]:

```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=30)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 43s 6ms/step - loss: 1.3452 - acc: 0.4075 - val_loss: 1.24
81 - val_acc: 0.4221
Epoch 2/30
7352/7352 [==============================] - 43s 6ms/step - loss: 1.0971 - acc: 0.5188 - val_loss: 0.95
94 - val_acc: 0.5864
Epoch 3/30
7352/7352 [==============================] - 43s 6ms/step - loss: 0.9028 - acc: 0.5902 - val_loss: 0.93
64 - val_acc: 0.6020
Epoch 4/30
7352/7352 [==============================] - 38s 5ms/step - loss: 1.0618 - acc: 0.5427 - val_loss: 0.94
05 - val_acc: 0.5809
Epoch 5/30
7352/7352 [==============================] - 38s 5ms/step - loss: 0.7319 - acc: 0.6440 - val_loss: 0.89
35 - val_acc: 0.6084
Epoch 6/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.7305 - acc: 0.6556 - val_loss: 0.99
54 - val_acc: 0.6379
Epoch 7/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.7126 - acc: 0.6755 - val_loss: 0.81
36 - val_acc: 0.6709
Epoch 8/30
7352/7352 [==============================] - 38s 5ms/step - loss: 0.6725 - acc: 0.7074 - val_loss: 0.74
31 - val_acc: 0.6956
Epoch 9/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.5744 - acc: 0.7501 - val_loss: 0.70
13 - val_acc: 0.7452
Epoch 10/30
7352/7352 [==============================] - 39s 5ms/step - loss: 0.7282 - acc: 0.7405 - val_loss: 0.58
54 - val_acc: 0.7957
Epoch 11/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.4907 - acc: 0.8279 - val_loss: 0.59
03 - val_acc: 0.8310
Epoch 12/30
7352/7352 [==============================] - 39s 5ms/step - loss: 0.4096 - acc: 0.8734 - val_loss: 0.50
65 - val_acc: 0.8480
Epoch 13/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.3250 - acc: 0.9008 - val_loss: 0.52
46 - val_acc: 0.8622
Epoch 14/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.3042 - acc: 0.9066 - val_loss: 0.58
01 - val_acc: 0.8697
Epoch 15/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.2702 - acc: 0.9115 - val_loss: 0.33
95 - val_acc: 0.8911
Epoch 16/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.2494 - acc: 0.9242 - val_loss: 0.36
68 - val_acc: 0.8938
Epoch 17/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.2278 - acc: 0.9276 - val_loss: 0.50
20 - val_acc: 0.8904
Epoch 18/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.2344 - acc: 0.9232 - val_loss: 0.45
```

```
49 - val_acc: 0.8734
Epoch 19/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.2237 - acc: 0.9314 - val_loss: 0.40
07 - val_acc: 0.8877
Epoch 20/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.2039 - acc: 0.9314 - val_loss: 0.42
28 - val_acc: 0.8975
Epoch 21/30
7352/7352 [==============================] - 38s 5ms/step - loss: 0.2164 - acc: 0.9289 - val_loss: 0.40
99 - val_acc: 0.8914
Epoch 22/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.2166 - acc: 0.9298 - val_loss: 0.66
49 - val_acc: 0.8677
Epoch 23/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1829 - acc: 0.9389 - val_loss: 0.60
77 - val_acc: 0.8870
Epoch 24/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.1826 - acc: 0.9400 - val_loss: 0.48
94 - val_acc: 0.8884
Epoch 25/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1882 - acc: 0.9381 - val_loss: 0.38
88 - val_acc: 0.8972
Epoch 26/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.1937 - acc: 0.9378 - val_loss: 0.48
02 - val_acc: 0.9033
Epoch 27/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1910 - acc: 0.9395 - val_loss: 0.44
24 - val_acc: 0.9046
Epoch 28/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.1924 - acc: 0.9418 - val_loss: 0.56
22 - val_acc: 0.8860
Epoch 29/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1826 - acc: 0.9422 - val_loss: 0.47
24 - val_acc: 0.8826
Epoch 30/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.1747 - acc: 0.9410 - val_loss: 0.36
28 - val_acc: 0.8999
```

Out[67]:

```
<keras.callbacks.History at 0x21554934c18>
```

In [68]:

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

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 510        0        27        0                   0
SITTING                  2      368       121        0                   0
STANDING                 0       60       472        0                   0
WALKING                  0        1         0      462                  13
WALKING_DOWNSTAIRS       0        0         0        8                 401
WALKING_UPSTAIRS         4        2         1       17                   8

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            0
STANDING                           0
WALKING                           20
WALKING_DOWNSTAIRS                11
WALKING_UPSTAIRS                 439
```

In [69]:

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

```
2947/2947 [==============================] - 2s 687us/step
```

Out[69]:

Out[69]:

```
[0.3628289201868073, 0.8998982015609094]
```

In [54]:

```
out_table.add_row(["LSTM One - Highdata + sigmoid + 30 epoch","64","70%",score])
print(out_table)
```

```
+-----------------------------------------+-------------+---------+-----------------------------------
--------+
|                  MODEL                  | Hidden unit | Dropout |                  Accurcy
|
+-----------------------------------------+-------------+---------+-----------------------------------
--------+
|                LSTM One                 |      32     |   50%   |          [0.6969475907087326, 0.6]
|
|                LSTM One                 |      64     |   50%   |         [1.0818304223418236, 0.566]
|
|                LSTM One                 |      64     |   70%   |         [0.9148820471763611, 0.804]
|
|                LSTM Two                 |      64     |   80%   |         [1.1371510267443954, 0.578]
|
|           LSTM One - Highdata           |      64     |   80%   |           [nan, 0.168306752629793]
|
|        LSTM One - Highdata + Relu       |      64     |   70%   |           [nan, 0.168306752629793]
|
|      LSTM One - Highdata + sigmoid      |      64     |   70%   | [0.43612632637826976, 0.9022734984
730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |      64     |   70%   |  [0.3628289201868073, 0.8998982015
609094] |
+-----------------------------------------+-------------+---------+-----------------------------------
--------+
```

Our performance still has not improved dramatically. We have tries to find the number of layers and hidden units less data. As it did not work perfectly we will work with larger amount of data and 2 LSTM units. We will use 2 LSTM unit with 32 hidden units ith 70% dropout rate.

4th model

In [71]:

```
n_hidden = 32
```

In [72]:

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

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_18 (LSTM)               (None, 128, 32)           5376
_________________________________________________________________
dropout_14 (Dropout)         (None, 128, 32)           0
_________________________________________________________________
lstm_19 (LSTM)               (None, 32)                8320
_________________________________________________________________
dropout_15 (Dropout)         (None, 32)                0
```

```
_________________________________________________________________
dense_9 (Dense)              (None, 6)                 198       
=================================================================
Total params: 13,894
Trainable params: 13,894
Non-trainable params: 0
_________________________________________________________________
```

In [73]:

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

In [74]:

```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=30)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 68s 9ms/step - loss: 1.4068 - acc: 0.4157 - val_loss: 1.11
77 - val_acc: 0.5429
Epoch 2/30
7352/7352 [==============================] - 65s 9ms/step - loss: 1.0668 - acc: 0.5343 - val_loss: 0.96
37 - val_acc: 0.5663
Epoch 3/30
7352/7352 [==============================] - 68s 9ms/step - loss: 0.9339 - acc: 0.5660 - val_loss: 0.80
78 - val_acc: 0.6101
Epoch 4/30
7352/7352 [==============================] - 65s 9ms/step - loss: 0.8696 - acc: 0.6066 - val_loss: 0.89
73 - val_acc: 0.6043
Epoch 5/30
7352/7352 [==============================] - 70s 9ms/step - loss: 0.8696 - acc: 0.6065 - val_loss: 0.78
84 - val_acc: 0.6094
Epoch 6/30
7352/7352 [==============================] - 66s 9ms/step - loss: 0.8295 - acc: 0.6051 - val_loss: 0.76
85 - val_acc: 0.6067
Epoch 7/30
7352/7352 [==============================] - 65s 9ms/step - loss: 0.8018 - acc: 0.6230 - val_loss: 0.75
89 - val_acc: 0.6169
Epoch 8/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.7805 - acc: 0.6315 - val_loss: 0.78
62 - val_acc: 0.6142
Epoch 9/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.7717 - acc: 0.6352 - val_loss: 0.78
04 - val_acc: 0.6257
Epoch 10/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.7628 - acc: 0.6231 - val_loss: 0.76
69 - val_acc: 0.6267
Epoch 11/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.7567 - acc: 0.6379 - val_loss: 0.82
96 - val_acc: 0.6495
Epoch 12/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.7556 - acc: 0.6428 - val_loss: 0.80
44 - val_acc: 0.6837
Epoch 13/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.7312 - acc: 0.6493 - val_loss: 1.00
07 - val_acc: 0.6064
Epoch 14/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.7220 - acc: 0.6542 - val_loss: 0.77
92 - val_acc: 0.6770
Epoch 15/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.7351 - acc: 0.6510 - val_loss: 0.77
92 - val_acc: 0.6854
Epoch 16/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.7128 - acc: 0.6608 - val_loss: 0.84
93 - val_acc: 0.6922
Epoch 17/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.6929 - acc: 0.6889 - val_loss: 0.75
```

```
46 - val_acc: 0.7075
Epoch 18/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.6725 - acc: 0.6862 - val_loss: 0.92
99 - val_acc: 0.7004
Epoch 19/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.6848 - acc: 0.7093 - val_loss: 0.67
20 - val_acc: 0.7540
Epoch 20/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.6193 - acc: 0.7364 - val_loss: 0.74
18 - val_acc: 0.7818
Epoch 21/30
7352/7352 [==============================] - 66s 9ms/step - loss: 0.5843 - acc: 0.7712 - val_loss: 0.56
87 - val_acc: 0.8246s: 0.5856 - acc: 0
Epoch 22/30
7352/7352 [==============================] - 71s 10ms/step - loss: 0.5370 - acc: 0.7984 - val_loss: 0.5
811 - val_acc: 0.8517
Epoch 23/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.5141 - acc: 0.8215 - val_loss: 0.47
61 - val_acc: 0.8728
Epoch 24/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.4696 - acc: 0.8447 - val_loss: 0.53
22 - val_acc: 0.8660
Epoch 25/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.4607 - acc: 0.8520 - val_loss: 0.58
35 - val_acc: 0.8707
Epoch 26/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.4987 - acc: 0.8555 - val_loss: 0.51
61 - val_acc: 0.8867
Epoch 27/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.4225 - acc: 0.8671 - val_loss: 0.52
39 - val_acc: 0.8768
Epoch 28/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.4535 - acc: 0.8624 - val_loss: 0.49
24 - val_acc: 0.8806
Epoch 29/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.4009 - acc: 0.8758 - val_loss: 0.62
35 - val_acc: 0.8633
Epoch 30/30
7352/7352 [==============================] - 73s 10ms/step - loss: 0.3742 - acc: 0.8781 - val_loss: 0.5
005 - val_acc: 0.8856
```

Out[74]:

```
<keras.callbacks.History at 0x215578bc278>
```

In [75]:

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

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 510        0         0        0                   0
SITTING                  5      372        99        0                   0
STANDING                 0       76       453        0                   0
WALKING                  0        2         0      440                  33
WALKING_DOWNSTAIRS       0        1         0        8                 411
WALKING_UPSTAIRS         0        5         0       33                   9

Pred                WALKING_UPSTAIRS
True
LAYING                            27
SITTING                           15
STANDING                           3
WALKING                           21
WALKING_DOWNSTAIRS                 0
WALKING_UPSTAIRS                 424
```

In [76]:

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

```
2947/2947 [==============================] - 2s 847us/step
```

Out[76]:

```
[0.5005468663244095, 0.8856464200882254]
```

In [57]:

```
out_table.add_row(["LSTM Two - Highdata","32","70%",score])
print(out_table)
```

```
+----------------------------------------+-------------+---------+-----------------------------------
--------+
|                 MODEL                  | Hidden unit | Dropout |                 Accurcy
|
+----------------------------------------+-------------+---------+-----------------------------------
--------+
|               LSTM One                 |      32     |   50%   |         [0.6969475907087326, 0.6]
|
|               LSTM One                 |      64     |   50%   |        [1.0818304223418236, 0.566]
|
|               LSTM One                 |      64     |   70%   |        [0.9148820471763611, 0.804]
|
|               LSTM Two                 |      64     |   80%   |        [1.1371510267443954, 0.578]
|
|          LSTM One - Highdata           |      64     |   80%   |          [nan, 0.168306752629793]
|
|       LSTM One - Highdata + Relu       |      64     |   70%   |          [nan, 0.168306752629793]
|
|     LSTM One - Highdata + sigmoid      |      64     |   70%   | [0.43612632637826976, 0.9022734984
730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |      64     |   70%   |  [0.3628289201868073, 0.8998982015
609094] |
|          LSTM Two - Highdata           |      32     |   70%   |  [0.5005468663244095, 0.8856464200
882254] |
+----------------------------------------+-------------+---------+-----------------------------------
--------+
```

No Significant improvement with Two LSTM layers.

We have observed Accuracy improvement by increasing dropout rate while we were working with lesser data. But with large data high dropout rate hits the problem of over fitting.

We will try to build model with different hidden units and dropout rates in different layer

Model 6

In [78]:

```
n_hidden1 = 128
n_hidden2 =64
```

In [79]:

```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(n_hidden1, input_shape=(timesteps, input_dim), return_sequences=True))
# Adding a dropout layer
model.add(Dropout(0.5))
# Configuring the parameters for second layer
model.add(LSTM(n_hidden2))
# 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()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
```

```
Layer (type)                 Output Shape              Param #
=================================================================
lstm_20 (LSTM)               (None, 128, 128)          70656
_________________________________________________________________
dropout_16 (Dropout)         (None, 128, 128)          0
_________________________________________________________________
lstm_21 (LSTM)               (None, 64)                49408
_________________________________________________________________
dropout_17 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_10 (Dense)             (None, 6)                 390
=================================================================
Total params: 120,454
Trainable params: 120,454
Non-trainable params: 0
_________________________________________________________________
```

In [80]:

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

In [81]:

```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=30)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 119s 16ms/step - loss: 1.2797 - acc: 0.4528 - val_loss: 0.
9229 - val_acc: 0.5989
Epoch 2/30
7352/7352 [==============================] - 126s 17ms/step - loss: 0.7741 - acc: 0.6253 - val_loss: 0.
7532 - val_acc: 0.6013
Epoch 3/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.7068 - acc: 0.6508 - val_loss: 0.
7633 - val_acc: 0.6471
Epoch 4/30
7352/7352 [==============================] - 122s 17ms/step - loss: 0.7289 - acc: 0.6477 - val_loss: 2.
0970 - val_acc: 0.4421
Epoch 5/30
7352/7352 [==============================] - 118s 16ms/step - loss: 0.9533 - acc: 0.5847 - val_loss: 0.
7254 - val_acc: 0.6298
Epoch 6/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.7065 - acc: 0.6532 - val_loss: 0.
6935 - val_acc: 0.6715
Epoch 7/30
7352/7352 [==============================] - 120s 16ms/step - loss: 0.6741 - acc: 0.6704 - val_loss: 0.
6757 - val_acc: 0.6960
Epoch 8/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.6246 - acc: 0.7164 - val_loss: 0.
4909 - val_acc: 0.8259
Epoch 9/30
7352/7352 [==============================] - 119s 16ms/step - loss: 0.5184 - acc: 0.7973 - val_loss: 0.
4183 - val_acc: 0.8592
Epoch 10/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.3467 - acc: 0.8979 - val_loss: 0.
4233 - val_acc: 0.8544
Epoch 11/30
7352/7352 [==============================] - 120s 16ms/step - loss: 0.2675 - acc: 0.9192 - val_loss: 0.
3873 - val_acc: 0.8904
Epoch 12/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.2332 - acc: 0.9300 - val_loss: 0.
3849 - val_acc: 0.9036
Epoch 13/30
7352/7352 [==============================] - 119s 16ms/step - loss: 0.2036 - acc: 0.9358 - val_loss: 0.
3795 - val_acc: 0.9070
Epoch 14/30
7352/7352 [==============================] - 118s 16ms/step - loss: 0.1930 - acc: 0.9396 - val_loss: 0.
```

```
7352/7352 [==============================] - 118s 16ms/step - loss: 0.1950 - acc: 0.9396 - val_loss: 0.
3107 - val_acc: 0.9013
Epoch 15/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.1880 - acc: 0.9408 - val_loss: 0.
4066 - val_acc: 0.8985
Epoch 16/30
7352/7352 [==============================] - 113s 15ms/step - loss: 0.1609 - acc: 0.9404 - val_loss: 0.
4261 - val_acc: 0.8996
Epoch 17/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.1791 - acc: 0.9416 - val_loss: 0.
4510 - val_acc: 0.8972
Epoch 18/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.1729 - acc: 0.9471 - val_loss: 0.
4544 - val_acc: 0.9074
Epoch 19/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.2119 - acc: 0.9408 - val_loss: 0.
3160 - val_acc: 0.9023
Epoch 20/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.1636 - acc: 0.9456 - val_loss: 0.
5826 - val_acc: 0.8996
Epoch 21/30
7352/7352 [==============================] - 116s 16ms/step - loss: 0.3124 - acc: 0.9291 - val_loss: 0.
3749 - val_acc: 0.9091
Epoch 22/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.1682 - acc: 0.9412 - val_loss: 0.
5432 - val_acc: 0.8863
Epoch 23/30
7352/7352 [==============================] - 120s 16ms/step - loss: 0.1627 - acc: 0.9464 - val_loss: 0.
7289 - val_acc: 0.8867
Epoch 24/30
7352/7352 [==============================] - 116s 16ms/step - loss: 0.1595 - acc: 0.9472 - val_loss: 0.
5858 - val_acc: 0.9013
Epoch 25/30
7352/7352 [==============================] - 112s 15ms/step - loss: 0.1600 - acc: 0.9480 - val_loss: 0.
5505 - val_acc: 0.8985
Epoch 26/30
7352/7352 [==============================] - 116s 16ms/step - loss: 0.2302 - acc: 0.9362 - val_loss: 0.
6763 - val_acc: 0.8958
Epoch 27/30
7352/7352 [==============================] - 116s 16ms/step - loss: 0.1390 - acc: 0.9506 - val_loss: 0.
4488 - val_acc: 0.9196
Epoch 28/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.1436 - acc: 0.9518 - val_loss: 0.
5683 - val_acc: 0.9155
Epoch 29/30
7352/7352 [==============================] - 111s 15ms/step - loss: 0.1480 - acc: 0.9483 - val_loss: 0.
3974 - val_acc: 0.9196
Epoch 30/30
7352/7352 [==============================] - 112s 15ms/step - loss: nan - acc: 0.6348 - val_loss: nan -
val_acc: 0.1683
```

Out[81]:

```
<keras.callbacks.History at 0x21558a3e390>
```

In [82]:

```
# 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 [83]:

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

```
2947/2947 [==============================] - 8s 3ms/step
```

Out[83]:

```
[nan, 0.168306752629793]
```

In [58]:

```
out_table.add_row(["LSTM Two - Highdata","128 - 64","50% - 70%",score])
print(out_table)
```

```
+-----------------------------------------+-------------+-----------+-----------------------------------
----------+
|                  MODEL                  | Hidden unit |  Dropout  |                  Accurcy
|
+-----------------------------------------+-------------+-----------+-----------------------------------
----------+
|                LSTM One                 |      32     |    50%    |          [0.6969475907087326, 0.6
]         |
|                LSTM One                 |      64     |    50%    |         [1.0818304223418236, 0.56
6]         |
|                LSTM One                 |      64     |    70%    |         [0.9148820471763611, 0.80
4]         |
|                LSTM Two                 |      64     |    80%    |         [1.1371510267443954, 0.57
8]         |
|           LSTM One - Highdata           |      64     |    80%    |           [nan, 0.168306752629793
]         |
|        LSTM One - Highdata + Relu       |      64     |    70%    |           [nan, 0.168306752629793
]         |
|      LSTM One - Highdata + sigmoid      |      64     |    70%    | [0.43612632637826976, 0.90227349
84730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |      64     |    70%    |  [0.3628289201868073, 0.89989820
15609094] |
|           LSTM Two - Highdata           |      32     |    70%    |  [0.5005468663244095, 0.88564642
00882254] |
|           LSTM Two - Highdata           |   128 - 64  | 50% - 70% |           [nan, 0.168306752629793
]         |
+-----------------------------------------+-------------+-----------+-----------------------------------
----------+
```

As we can see the Model was performing well just before 30th epoch but just during the last epoch it got overfitted, we will try to reduce the Dropout little bit and try to run the model again

Model 7

In [85]:

```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(n_hidden1, input_shape=(timesteps, input_dim), return_sequences=True))
# Adding a dropout layer
model.add(Dropout(0.3))
# Configuring the parameters for second layer
model.add(LSTM(n_hidden2))
# 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()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_22 (LSTM)               (None, 128, 128)          70656
_________________________________________________________________
dropout_18 (Dropout)         (None, 128, 128)          0
_________________________________________________________________
lstm_23 (LSTM)               (None, 64)                49408
_________________________________________________________________
```

```
dropout_19 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_11 (Dense)             (None, 6)                 390
=================================================================
Total params: 120,454
Trainable params: 120,454
Non-trainable params: 0
_________________________________________________________________
```

In [86]:

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

In [87]:

```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=30)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 140s 19ms/step - loss: 1.1751 - acc: 0.5044 - val_loss: 0.
8747 - val_acc: 0.6542
Epoch 2/30
7352/7352 [==============================] - 126s 17ms/step - loss: 0.7740 - acc: 0.6496 - val_loss: 0.
6567 - val_acc: 0.7757
Epoch 3/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.7145 - acc: 0.6504 - val_loss: 0.
9938 - val_acc: 0.6305
Epoch 4/30
7352/7352 [==============================] - 116s 16ms/step - loss: 0.5627 - acc: 0.7768 - val_loss: 0.
4858 - val_acc: 0.8388
Epoch 5/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.4611 - acc: 0.8400 - val_loss: 0.
5290 - val_acc: 0.8090
Epoch 6/30
7352/7352 [==============================] - 116s 16ms/step - loss: 0.3396 - acc: 0.8904 - val_loss: 0.
4291 - val_acc: 0.8789
Epoch 7/30
7352/7352 [==============================] - 123s 17ms/step - loss: 0.2740 - acc: 0.9102 - val_loss: 0.
7283 - val_acc: 0.8239
Epoch 8/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.2608 - acc: 0.9085 - val_loss: 0.
3271 - val_acc: 0.9030
Epoch 9/30
7352/7352 [==============================] - 114s 16ms/step - loss: 0.1857 - acc: 0.9358 - val_loss: 0.
4573 - val_acc: 0.8731
Epoch 10/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.1654 - acc: 0.9380 - val_loss: 0.
2990 - val_acc: 0.9155
Epoch 11/30
7352/7352 [==============================] - 118s 16ms/step - loss: 0.1674 - acc: 0.9425 - val_loss: 0.
2691 - val_acc: 0.9216
Epoch 12/30
7352/7352 [==============================] - 114s 15ms/step - loss: 0.1463 - acc: 0.9457 - val_loss: 0.
4439 - val_acc: 0.8887
Epoch 13/30
7352/7352 [==============================] - 119s 16ms/step - loss: 0.1535 - acc: 0.9448 - val_loss: 0.
4222 - val_acc: 0.9080
Epoch 14/30
7352/7352 [==============================] - 116s 16ms/step - loss: 0.1961 - acc: 0.9396 - val_loss: 0.
3137 - val_acc: 0.9165
Epoch 15/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.1481 - acc: 0.9472 - val_loss: 0.
3113 - val_acc: 0.9101
Epoch 16/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.1397 - acc: 0.9442 - val_loss: 0.
3831 - val_acc: 0.8975
```

```
Epoch 17/30
7352/7352 [==============================] - 117s 16ms/step - loss: 0.1630 - acc: 0.9415 - val_loss: 0.
3474 - val_acc: 0.9080
Epoch 18/30
7352/7352 [==============================] - 116s 16ms/step - loss: 0.1368 - acc: 0.9501 - val_loss: 0.
4101 - val_acc: 0.9131
Epoch 19/30
7352/7352 [==============================] - 122s 17ms/step - loss: 0.1648 - acc: 0.9475 - val_loss: 0.
3551 - val_acc: 0.9155
Epoch 20/30
7352/7352 [==============================] - 119s 16ms/step - loss: 0.1298 - acc: 0.9512 - val_loss: 0.
4729 - val_acc: 0.8989
Epoch 21/30
7352/7352 [==============================] - 114s 16ms/step - loss: 0.1376 - acc: 0.9482 - val_loss: 0.
4968 - val_acc: 0.9053
Epoch 22/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.1953 - acc: 0.9319 - val_loss: 0.
5339 - val_acc: 0.8918
Epoch 23/30
7352/7352 [==============================] - 116s 16ms/step - loss: 0.1556 - acc: 0.9431 - val_loss: 0.
4169 - val_acc: 0.9284
Epoch 24/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.1875 - acc: 0.9363 - val_loss: 0.
4402 - val_acc: 0.9141
Epoch 25/30
7352/7352 [==============================] - 114s 15ms/step - loss: 0.1352 - acc: 0.9497 - val_loss: 0.
3714 - val_acc: 0.9175
Epoch 26/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.1684 - acc: 0.9403 - val_loss: 0.
4736 - val_acc: 0.9057
Epoch 27/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.1580 - acc: 0.9459 - val_loss: 0.
3666 - val_acc: 0.9155
Epoch 28/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.1696 - acc: 0.9431 - val_loss: 0.
3912 - val_acc: 0.9226
Epoch 29/30
7352/7352 [==============================] - 115s 16ms/step - loss: 0.1580 - acc: 0.9479 - val_loss: 0.
3716 - val_acc: 0.8935
Epoch 30/30
7352/7352 [==============================] - 116s 16ms/step - loss: 0.1521 - acc: 0.9495 - val_loss: 0.
3770 - val_acc: 0.9179
```

Out[87]:

```
<keras.callbacks.History at 0x2155d0086d8>
```

In [88]:

```
# 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      378       112        0                   0
STANDING                 0       79       453        0                   0
WALKING                  0        2         1      451                  33
WALKING_DOWNSTAIRS       0        0         1        0                 419
WALKING_UPSTAIRS         0        0         0        3                   1

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            1
STANDING                           0
WALKING                            9
WALKING_DOWNSTAIRS                 0
WALKING_UPSTAIRS                 467
```

In [89]:

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

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

Out[89]:

```
[0.3769736843092622, 0.9178825924669155]
```

In [61]:

```
out_table.add_row(["LSTM Two - Highdata","128 - 64","30% - 50%",score])
print(out_table)
```

```
+-----------------------------------------+-------------+-----------+-----------------------------------
----------+
|                  MODEL                  | Hidden unit |  Dropout  |                  Accurcy
|
+-----------------------------------------+-------------+-----------+-----------------------------------
----------+
|                LSTM One                 |      32     |    50%    |          [0.6969475907087326, 0.6
]          |
|                LSTM One                 |      64     |    50%    |         [1.0818304223418236, 0.56
6]         |
|                LSTM One                 |      64     |    70%    |         [0.9148820471763611, 0.80
4]         |
|                LSTM Two                 |      64     |    80%    |         [1.1371510267443954, 0.57
8]         |
|          LSTM One - Highdata            |      64     |    80%    |           [nan, 0.168306752629793
]          |
|        LSTM One - Highdata + Relu       |      64     |    70%    |           [nan, 0.168306752629793
]          |
|      LSTM One - Highdata + sigmoid      |      64     |    70%    | [0.43612632637826976, 0.90227349
84730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |      64     |    70%    |  [0.3628289201868073, 0.89989820
15609094] |
|          LSTM Two - Highdata            |      32     |    70%    |  [0.5005468663244095, 0.88564642
00882254] |
|          LSTM Two - Highdata            |   128 - 64  | 50% - 70% |           [nan, 0.168306752629793
]          |
|          LSTM Two - Highdata            |   128 - 64  | 30% - 50% |  [0.3769736843092622, 0.91788259
24669155] |
+-----------------------------------------+-------------+-----------+-----------------------------------
----------+
```

Conclusions

- The hardest problem we face here is distinguish between standing and sitting
- We can observe our confusin matrix to conclude the above
- We faced various problems but Using two LSTM and proper dropout we improved our Accuracy little bit

We Will try to use Conv1D and observe the performance

CNN Model1

In [39]:

```
import keras
from keras.layers import Conv2D, MaxPooling2D,Flatten,Conv1D
from keras.layers.convolutional import MaxPooling1D
```

In [64]:

```
model = Sequential()
model.add(Conv1D(32, kernel_size=3,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(Conv1D(64, (3,), activation='relu'))
```

```
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(n_classes, activation='softmax'))

model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_9 (Conv1D)            (None, 126, 32)           896
_________________________________________________________________
conv1d_10 (Conv1D)           (None, 124, 64)           6208
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 62, 64)            0
_________________________________________________________________
flatten_5 (Flatten)          (None, 3968)              0
_________________________________________________________________
dense_9 (Dense)              (None, 128)               508032
_________________________________________________________________
dropout_10 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_10 (Dense)             (None, 6)                 774
=================================================================
Total params: 515,910
Trainable params: 515,910
Non-trainable params: 0
_________________________________________________________________
```

In [65]:

```
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=15,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/15
7352/7352 [==============================] - 10s 1ms/step - loss: 0.5626 - acc: 0.7696 - val_loss: 0.63
82 - val_acc: 0.8412
Epoch 2/15
7352/7352 [==============================] - 10s 1ms/step - loss: 0.2259 - acc: 0.9159 - val_loss: 0.49
97 - val_acc: 0.8823
Epoch 3/15
7352/7352 [==============================] - 13s 2ms/step - loss: 0.1762 - acc: 0.9334 - val_loss: 0.42
19 - val_acc: 0.8670
Epoch 4/15
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1536 - acc: 0.9407 - val_loss: 0.45
92 - val_acc: 0.8958
Epoch 5/15
7352/7352 [==============================] - 9s 1ms/step - loss: 0.1617 - acc: 0.9399 - val_loss: 0.266
9 - val_acc: 0.9091
Epoch 6/15
7352/7352 [==============================] - 9s 1ms/step - loss: 0.1312 - acc: 0.9455 - val_loss: 0.396
7 - val_acc: 0.8955
Epoch 7/15
7352/7352 [==============================] - 9s 1ms/step - loss: 0.1191 - acc: 0.9460 - val_loss: 0.555
8 - val_acc: 0.8965
Epoch 8/15
7352/7352 [==============================] - 9s 1ms/step - loss: 0.1233 - acc: 0.9472 - val_loss: 0.466
2 - val_acc: 0.8948
Epoch 9/15
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1337 - acc: 0.9470 - val_loss: 0.48
18 - val_acc: 0.9040
Epoch 10/15
```

```
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1160 - acc: 0.9502 - val_loss: 0.55
83 - val_acc: 0.9046
Epoch 11/15
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1467 - acc: 0.9456 - val_loss: 0.43
09 - val_acc: 0.8996
Epoch 12/15
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1178 - acc: 0.9513 - val_loss: 0.43
85 - val_acc: 0.9094
Epoch 13/15
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1065 - acc: 0.9518 - val_loss: 0.56
68 - val_acc: 0.8873
Epoch 14/15
7352/7352 [==============================] - 9s 1ms/step - loss: 0.1059 - acc: 0.9518 - val_loss: 0.426
6 - val_acc: 0.9145
Epoch 15/15
7352/7352 [==============================] - 9s 1ms/step - loss: 0.1009 - acc: 0.9547 - val_loss: 0.616
3 - val_acc: 0.9077
Test loss: 0.6163437596247316
Test accuracy: 0.9077027485578555
```

In [66]:

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

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 510        0         0        0                   0
SITTING                  0      417        49        0                   0
STANDING                 0       98       432        1                   0
WALKING                  0        0         0      460                  33
WALKING_DOWNSTAIRS       0        0         0        2                 416
WALKING_UPSTAIRS         0        0         0        3                  28

Pred                WALKING_UPSTAIRS
True
LAYING                            27
SITTING                           25
STANDING                           1
WALKING                            3
WALKING_DOWNSTAIRS                 2
WALKING_UPSTAIRS                 440
```

In [67]:

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

```
2947/2947 [==============================] - 1s 174us/step
```

Out[67]:

```
[0.6163437596247316, 0.9077027485578555]
```

In [68]:

```
out_table.add_row(["CNN","32 - 64","50%",score])
print(out_table)
```

```
+----------------------------------------+-------------+-----------+--------------------------------
----------+
|                 MODEL                  | Hidden unit |  Dropout  |                 Accurcy
|
+----------------------------------------+-------------+-----------+--------------------------------
----------+
|               LSTM One                 |      32     |    50%    |         [0.6969475907087326, 0.6
]         |
|               LSTM One                 |      64     |    50%    |        [1.0818304223418236, 0.56
6]        |
|               LSTM One                 |      64     |    70%    |        [0.9148820471763611, 0.80
4]        |
|               LSTM Two                 |      64     |    80%    |        [1.1371510267443954, 0.57
```

```
8]        |
|           LSTM One - Highdata            |      64     |    80%    |          [nan, 0.168306752629793
]         |
|        LSTM One - Highdata + Relu        |      64     |    70%    |          [nan, 0.168306752629793
]         |
|      LSTM One - Highdata + sigmoid       |      64     |    70%    | [0.43612632637826976, 0.90227349
84730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |      64     |    70%    |  [0.3628289201868073, 0.89989820
15609094] |
|           LSTM Two - Highdata            |      32     |    70%    |  [0.5005468663244095, 0.88564642
00882254] |
|           LSTM Two - Highdata            |   128 - 64  | 50% - 70% |          [nan, 0.168306752629793
]         |
|           LSTM Two - Highdata            |   128 - 64  | 30% - 50% |  [0.3769736843092622, 0.91788259
24669155] |
|                  CNN                     |   32 - 64   |    50%    |  [0.6163437596247316, 0.90770274
85578555] |
+------------------------------------------+-------------+-----------+-----------------------------------
----------+
```

CNN Model2

In [69]:

```
model = Sequential()
model.add(Conv1D(64, kernel_size=3,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(Conv1D(128, 3, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(n_classes, activation='softmax'))

model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_11 (Conv1D)           (None, 126, 64)           1792
_________________________________________________________________
conv1d_12 (Conv1D)           (None, 124, 128)          24704
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 62, 128)           0
_________________________________________________________________
flatten_6 (Flatten)          (None, 7936)              0
_________________________________________________________________
dense_11 (Dense)             (None, 128)               1015936
_________________________________________________________________
dropout_11 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_12 (Dense)             (None, 6)                 774
=================================================================
Total params: 1,043,206
Trainable params: 1,043,206
Non-trainable params: 0
_________________________________________________________________
```

In [70]:

```
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=15,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
```

```
print('Test accuracy:', score[1])
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/15
7352/7352 [==============================] - 21s 3ms/step - loss: 0.5252 - acc: 0.7930 - val_loss: 0.43
89 - val_acc: 0.8778
Epoch 2/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.2023 - acc: 0.9207 - val_loss: 0.47
71 - val_acc: 0.8901
Epoch 3/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1821 - acc: 0.9351 - val_loss: 0.28
15 - val_acc: 0.9131
Epoch 4/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1577 - acc: 0.9387 - val_loss: 0.42
02 - val_acc: 0.8951
Epoch 5/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1614 - acc: 0.9422 - val_loss: 0.48
73 - val_acc: 0.8962
Epoch 6/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1362 - acc: 0.9455 - val_loss: 0.39
72 - val_acc: 0.9060
Epoch 7/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1177 - acc: 0.9513 - val_loss: 0.27
45 - val_acc: 0.9152
Epoch 8/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1230 - acc: 0.9516 - val_loss: 0.24
40 - val_acc: 0.9220
Epoch 9/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1275 - acc: 0.9470 - val_loss: 0.32
97 - val_acc: 0.9101
Epoch 10/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1208 - acc: 0.9491 - val_loss: 0.31
54 - val_acc: 0.9203
Epoch 11/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1344 - acc: 0.9491 - val_loss: 0.57
29 - val_acc: 0.8873
Epoch 12/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1496 - acc: 0.9479 - val_loss: 0.43
88 - val_acc: 0.9111
Epoch 13/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1369 - acc: 0.9516 - val_loss: 0.55
49 - val_acc: 0.9067
Epoch 14/15
7352/7352 [==============================] - 21s 3ms/step - loss: 0.1168 - acc: 0.9518 - val_loss: 0.64
71 - val_acc: 0.9111
Epoch 15/15
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1240 - acc: 0.9516 - val_loss: 0.45
08 - val_acc: 0.9169
Test loss: 0.4507699583859416
Test accuracy: 0.9168646080760094
```

In [71]:

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

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 520        0         0        0                   0
SITTING                  0      383       103        0                   0
STANDING                 0       78       453        0                   0
WALKING                  0        0         0      489                   4
WALKING_DOWNSTAIRS       0        0         0        3                 413
WALKING_UPSTAIRS         0        0         0        5                  22

Pred                WALKING_UPSTAIRS
True
LAYING                            17
SITTING                            5
STANDING                           1
WALKING                            3
WALKING_DOWNSTAIRS                 4
WALKING_UPSTAIRS                 444
```

In [72]:

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

```
2947/2947 [==============================] - 1s 421us/step
```

Out[72]:

```
[0.4507699583859416, 0.9168646080760094]
```

In [73]:

```
out_table.add_row(["CNN","64 - 128","50%",score])
print(out_table)
```

```
+-----------------------------------------+-------------+-----------+-----------------------------------
----------+
|                  MODEL                  | Hidden unit |  Dropout  |                  Accurcy
|
+-----------------------------------------+-------------+-----------+-----------------------------------
----------+
|                LSTM One                 |      32     |    50%    |          [0.6969475907087326, 0.6
]         |
|                LSTM One                 |      64     |    50%    |         [1.0818304223418236, 0.56
6]         |
|                LSTM One                 |      64     |    70%    |         [0.9148820471763611, 0.80
4]         |
|                LSTM Two                 |      64     |    80%    |         [1.1371510267443954, 0.57
8]         |
|           LSTM One - Highdata           |      64     |    80%    |           [nan, 0.168306752629793
]         |
|        LSTM One - Highdata + Relu       |      64     |    70%    |           [nan, 0.168306752629793
]         |
|      LSTM One - Highdata + sigmoid      |      64     |    70%    | [0.43612632637826976, 0.90227349
84730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |      64     |    70%    |  [0.3628289201868073, 0.89989820
15609094] |
|           LSTM Two - Highdata           |      32     |    70%    |  [0.5005468663244095, 0.88564642
00882254] |
|           LSTM Two - Highdata           |   128 - 64  | 50% - 70% |           [nan, 0.168306752629793
]         |
|           LSTM Two - Highdata           |   128 - 64  | 30% - 50% |  [0.3769736843092622, 0.91788259
24669155] |
|                  CNN                    |   32 - 64   |    50%    |  [0.6163437596247316, 0.90770274
85578555] |
|                  CNN                    |   64 - 128  |    50%    |  [0.4507699583859416, 0.91686460
80760094] |
+-----------------------------------------+-------------+-----------+-----------------------------------
----------+
```

Improved with increasing the filters. but not a remarkable improvement We will see by increasing the Kernal size

CNN Model 3

In [74]:

```
model = Sequential()
model.add(Conv1D(64, kernel_size=20,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(Conv1D(128, 20, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(n_classes, activation='softmax'))

model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_13 (Conv1D)           (None, 109, 64)           11584
_________________________________________________________________
conv1d_14 (Conv1D)           (None, 90, 128)           163968
_________________________________________________________________
max_pooling1d_4 (MaxPooling1 (None, 45, 128)           0
_________________________________________________________________
flatten_7 (Flatten)          (None, 5760)              0
_________________________________________________________________
dense_13 (Dense)             (None, 128)               737408
_________________________________________________________________
dropout_12 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_14 (Dense)             (None, 6)                 774
=================================================================
Total params: 913,734
Trainable params: 913,734
Non-trainable params: 0
_________________________________________________________________
```

In [75]:

```
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=15,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/15
7352/7352 [==============================] - 43s 6ms/step - loss: 0.4509 - acc: 0.8252 - val_loss: 0.23
60 - val_acc: 0.9101
Epoch 2/15
7352/7352 [==============================] - 40s 5ms/step - loss: 0.1916 - acc: 0.9325 - val_loss: 0.44
39 - val_acc: 0.9040
Epoch 3/15
7352/7352 [==============================] - 41s 6ms/step - loss: 0.1727 - acc: 0.9355 - val_loss: 0.38
79 - val_acc: 0.9226
Epoch 4/15
7352/7352 [==============================] - 40s 5ms/step - loss: 0.1760 - acc: 0.9426 - val_loss: 0.82
54 - val_acc: 0.8795
Epoch 5/15
7352/7352 [==============================] - 41s 6ms/step - loss: 0.3325 - acc: 0.9260 - val_loss: 0.28
77 - val_acc: 0.9209
Epoch 6/15
7352/7352 [==============================] - 41s 6ms/step - loss: 0.1649 - acc: 0.9422 - val_loss: 0.34
92 - val_acc: 0.9104
Epoch 7/15
7352/7352 [==============================] - 40s 5ms/step - loss: 0.1240 - acc: 0.9480 - val_loss: 0.33
78 - val_acc: 0.9192
Epoch 8/15
7352/7352 [==============================] - 41s 6ms/step - loss: 0.1227 - acc: 0.9497 - val_loss: 0.30
91 - val_acc: 0.9240
Epoch 9/15
7352/7352 [==============================] - 41s 6ms/step - loss: 0.1190 - acc: 0.9491 - val_loss: 0.29
97 - val_acc: 0.9152
Epoch 10/15
7352/7352 [==============================] - 41s 6ms/step - loss: 0.1422 - acc: 0.9457 - val_loss: 0.39
89 - val_acc: 0.9138
Epoch 11/15
7352/7352 [==============================] - 42s 6ms/step - loss: 0.1716 - acc: 0.9437 - val_loss: 0.43
87 - val_acc: 0.9108
Epoch 12/15
7352/7352 [==============================] - 43s 6ms/step - loss: 0.3734 - acc: 0.9335 - val_loss: 0.40
98 - val_acc: 0.9074
Epoch 13/15
```

```
7352/7352 [==============================] - 42s 6ms/step - loss: 0.2109 - acc: 0.9433 - val_loss: 0.44
87 - val_acc: 0.9121
Epoch 14/15
7352/7352 [==============================] - 41s 6ms/step - loss: 0.2154 - acc: 0.9459 - val_loss: 0.40
31 - val_acc: 0.9192
Epoch 15/15
7352/7352 [==============================] - 40s 5ms/step - loss: 0.1755 - acc: 0.9461 - val_loss: 0.48
55 - val_acc: 0.9175
Test loss: 0.48552406951467336
Test accuracy: 0.9175432643366135
```

In [76]:

```
# 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      342       147        0                   0
STANDING                 0       44       487        0                   0
WALKING                  1        0         0      466                  25
WALKING_DOWNSTAIRS       0        0         0        1                 419
WALKING_UPSTAIRS         0        0         0        0                  18

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            2
STANDING                           1
WALKING                            4
WALKING_DOWNSTAIRS                 0
WALKING_UPSTAIRS                 453
```

In [77]:

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

```
2947/2947 [==============================] - 4s 1ms/step
```

Out[77]:

```
[0.48552406951467336, 0.9175432643366135]
```

In [78]:

```
out_table.add_row(["CNN","64 - 128 kernal=20","50%",score])
print(out_table)
```

```
+-----------------------------------------+--------------------+-----------+--------------------------
-----------------+
|                  MODEL                  |    Hidden unit     |  Dropout  |                  Accurcy
|
+-----------------------------------------+--------------------+-----------+--------------------------
-----------------+
|                 LSTM One                |         32         |    50%    |          [0.69694759070873
26, 0.6]          |
|                 LSTM One                |         64         |    50%    |         [1.081830422341823
6, 0.566]         |
|                 LSTM One                |         64         |    70%    |         [0.914882047176361
1, 0.804]         |
|                 LSTM Two                |         64         |    80%    |         [1.137151026744395
4, 0.578]         |
|           LSTM One - Highdata           |         64         |    80%    |           [nan, 0.16830675
2629793]          |
|        LSTM One - Highdata + Relu       |         64         |    70%    |           [nan, 0.16830675
2629793]          |
|      LSTM One - Highdata + sigmoid      |         64         |    70%    | [0.43612632637826976, 0.9
022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |         64         |    70%    |  [0.3628289201868073, 0.8
```

```
99898201560909 4] |
|           LSTM Two - Highdata            |         32         |    70%    |  [0.5005468663244095, 0.8
856464200882254] |
|           LSTM Two - Highdata            |      128 - 64      | 50% - 70% |           [nan, 0.16830675
2629793]         |
|           LSTM Two - Highdata            |      128 - 64      | 30% - 50% |  [0.3769736843092622, 0.9
178825924669155] |
|                  CNN                     |      32 - 64       |    50%    |  [0.6163437596247316, 0.9
077027485578555] |
|                  CNN                     |      64 - 128      |    50%    |  [0.4507699583859416, 0.9
168646080760094] |
|                  CNN                     | 64 - 128 kernal=20 |    50%    | [0.48552406951467336, 0.9
175432643366135] |
+------------------------------------------+--------------------+-----------+-------------------------
-----------------+
```

Accuracy still doesnot seem enogh as our AIM is to get accuracy greater than 96%. We will try to add an extra convolution layer with every layer max pool capability

CNN Model4

In [80]:

```
model = Sequential()
model.add(Conv1D(32, kernel_size=3,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(64, 5, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(128, 5, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(n_classes, activation='softmax'))

model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_18 (Conv1D)           (None, 126, 32)           896
_________________________________________________________________
max_pooling1d_7 (MaxPooling1 (None, 63, 32)            0
_________________________________________________________________
conv1d_19 (Conv1D)           (None, 59, 64)            10304
_________________________________________________________________
max_pooling1d_8 (MaxPooling1 (None, 29, 64)            0
_________________________________________________________________
conv1d_20 (Conv1D)           (None, 25, 128)           41088
_________________________________________________________________
max_pooling1d_9 (MaxPooling1 (None, 12, 128)           0
_________________________________________________________________
flatten_8 (Flatten)          (None, 1536)              0
_________________________________________________________________
dense_15 (Dense)             (None, 128)               196736
_________________________________________________________________
dropout_13 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_16 (Dense)             (None, 6)                 774
=================================================================
Total params: 249,798
Trainable params: 249,798
Non-trainable params: 0
_________________________________________________________________
```

In [81]:

```
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])
```

```
history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=15,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/15
7352/7352 [==============================] - 9s 1ms/step - loss: 0.4601 - acc: 0.8081 - val_loss: 0.366
5 - val_acc: 0.8968
Epoch 2/15
7352/7352 [==============================] - 9s 1ms/step - loss: 0.1562 - acc: 0.9395 - val_loss: 0.446
0 - val_acc: 0.8982
Epoch 3/15
7352/7352 [==============================] - 9s 1ms/step - loss: 0.1394 - acc: 0.9438 - val_loss: 0.393
6 - val_acc: 0.8992
Epoch 4/15
7352/7352 [==============================] - 9s 1ms/step - loss: 0.1179 - acc: 0.9504 - val_loss: 0.385
7 - val_acc: 0.9189
Epoch 5/15
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1174 - acc: 0.9486 - val_loss: 0.381
8 - val_acc: 0.9199
Epoch 6/15
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1239 - acc: 0.9457 - val_loss: 0.439
3 - val_acc: 0.9019
Epoch 7/15
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1241 - acc: 0.9504 - val_loss: 0.409
0 - val_acc: 0.9040
Epoch 8/15
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1134 - acc: 0.9517 - val_loss: 0.352
6 - val_acc: 0.9220
Epoch 9/15
7352/7352 [==============================] - 8s 1ms/step - loss: 0.2061 - acc: 0.9453 - val_loss: 0.595
9 - val_acc: 0.9026
Epoch 10/15
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1537 - acc: 0.9482 - val_loss: 0.459
3 - val_acc: 0.9057
Epoch 11/15
7352/7352 [==============================] - 9s 1ms/step - loss: 0.1180 - acc: 0.9516 - val_loss: 0.569
4 - val_acc: 0.9043
Epoch 12/15
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1087 - acc: 0.9539 - val_loss: 0.501
2 - val_acc: 0.9063
Epoch 13/15
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1222 - acc: 0.9502 - val_loss: 0.517
4 - val_acc: 0.9226
Epoch 14/15
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1040 - acc: 0.9553 - val_loss: 0.515
6 - val_acc: 0.8945
Epoch 15/15
7352/7352 [==============================] - 8s 1ms/step - loss: 0.0982 - acc: 0.9570 - val_loss: 0.529
3 - val_acc: 0.9128
Test loss: 0.529261440275583
Test accuracy: 0.9127926705123854
```

In [82]:

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

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 510        0         0        0                   0
SITTING                  0      355       128        0                   0
STANDING                 0       49       482        0                   0
WALKING                  0        0         0      491                   4
WALKING_DOWNSTAIRS       0        0         0        0                 420
WALKING_UPSTAIRS         0        0         0        0                  39

Pred                WALKING_UPSTAIRS
True
```

```
True
LAYING                             27
SITTING                             8
STANDING                            1
WALKING                             1
WALKING_DOWNSTAIRS                  0
WALKING_UPSTAIRS                  432
```

In [83]:

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

```
2947/2947 [==============================] - 1s 208us/step
```

Out[83]:

```
[0.529261440275583, 0.9127926705123854]
```

In [86]:

```
out_table.add_row(["CNN","32-64-128 kernal=3,5,5","50%",score])
print(out_table)
```

```
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                  MODEL                  |      Hidden unit      |  Dropout  |                  Accu
rcy                   |
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                LSTM One                 |          32           |    50%    |          [0.6969475907
087326, 0.6]          |
|                LSTM One                 |          64           |    50%    |         [1.08183042234
18236, 0.566]         |
|                LSTM One                 |          64           |    70%    |         [0.91488204717
63611, 0.804]         |
|                LSTM Two                 |          64           |    80%    |         [1.13715102674
43954, 0.578]         |
|           LSTM One - Highdata           |          64           |    80%    |           [nan, 0.1683
06752629793]          |
|        LSTM One - Highdata + Relu       |          64           |    70%    |           [nan, 0.1683
06752629793]          |
|      LSTM One - Highdata + sigmoid      |          64           |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |          64           |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|           LSTM Two - Highdata           |          32           |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|           LSTM Two - Highdata           |        128 - 64       | 50% - 70% |           [nan, 0.1683
06752629793]          |
|           LSTM Two - Highdata           |        128 - 64       | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                  CNN                    |        32 - 64        |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                  CNN                    |        64 - 128       |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                  CNN                    |   64 - 128 kernal=20  |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                  CNN                    | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
```

CNN Model5

In [89]:

```
model = Sequential()
model.add(Conv1D(64, kernel_size=5,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
```

```
model.add(Conv1D(64, 7, activation='relu'))
model.add(Conv1D(128, 7, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(n_classes, activation='softmax'))

model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_27 (Conv1D)           (None, 124, 64)           2944
_________________________________________________________________
conv1d_28 (Conv1D)           (None, 118, 64)           28736
_________________________________________________________________
conv1d_29 (Conv1D)           (None, 112, 128)          57472
_________________________________________________________________
max_pooling1d_13 (MaxPooling (None, 28, 128)           0
_________________________________________________________________
flatten_10 (Flatten)         (None, 3584)              0
_________________________________________________________________
dense_19 (Dense)             (None, 128)               458880
_________________________________________________________________
dropout_15 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_20 (Dense)             (None, 6)                 774
=================================================================
Total params: 548,806
Trainable params: 548,806
Non-trainable params: 0
_________________________________________________________________
```

In [90]:

```
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 31s 4ms/step - loss: 0.4821 - acc: 0.8048 - val_loss: 0.77
36 - val_acc: 0.8521
Epoch 2/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.2068 - acc: 0.9248 - val_loss: 0.28
84 - val_acc: 0.9006
Epoch 3/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1480 - acc: 0.9414 - val_loss: 0.31
81 - val_acc: 0.9223
Epoch 4/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1553 - acc: 0.9456 - val_loss: 0.69
03 - val_acc: 0.8700
Epoch 5/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1900 - acc: 0.9395 - val_loss: 0.64
22 - val_acc: 0.8928
Epoch 6/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1595 - acc: 0.9441 - val_loss: 0.46
82 - val_acc: 0.9046
Epoch 7/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1183 - acc: 0.9527 - val_loss: 0.36
62 - val_acc: 0.9172
Epoch 8/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1428 - acc: 0.9475 - val_loss: 0.30
95 - val_acc: 0.9189
Epoch 9/30
```

```
Epoch 9/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1102 - acc: 0.9525 - val_loss: 0.41
15 - val_acc: 0.9114
Epoch 10/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1067 - acc: 0.9529 - val_loss: 0.48
35 - val_acc: 0.9162
Epoch 11/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1046 - acc: 0.9529 - val_loss: 0.39
99 - val_acc: 0.9216
Epoch 12/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1029 - acc: 0.9565 - val_loss: 0.48
35 - val_acc: 0.9186
Epoch 13/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.9662 - acc: 0.8940 - val_loss: 0.97
36 - val_acc: 0.8928
Epoch 14/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.6242 - acc: 0.9123 - val_loss: 0.79
60 - val_acc: 0.8948
Epoch 15/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.4225 - acc: 0.9279 - val_loss: 0.73
90 - val_acc: 0.9026
Epoch 16/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.2006 - acc: 0.9442 - val_loss: 0.49
29 - val_acc: 0.9131
Epoch 17/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1664 - acc: 0.9470 - val_loss: 0.50
03 - val_acc: 0.9209
Epoch 18/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1951 - acc: 0.9467 - val_loss: 0.62
04 - val_acc: 0.9074
Epoch 19/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.3248 - acc: 0.9359 - val_loss: 0.75
90 - val_acc: 0.8999
Epoch 20/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1589 - acc: 0.9516 - val_loss: 0.90
88 - val_acc: 0.9002
Epoch 21/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.6002 - acc: 0.9136 - val_loss: 0.72
00 - val_acc: 0.8985
Epoch 22/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1773 - acc: 0.9464 - val_loss: 0.77
98 - val_acc: 0.9060
Epoch 23/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.2564 - acc: 0.9441 - val_loss: 0.93
16 - val_acc: 0.8955
Epoch 24/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.3183 - acc: 0.9418 - val_loss: 0.70
02 - val_acc: 0.9057
Epoch 25/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.3855 - acc: 0.9338 - val_loss: 0.74
52 - val_acc: 0.9016
Epoch 26/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.3038 - acc: 0.9381 - val_loss: 0.88
03 - val_acc: 0.9036
Epoch 27/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.2599 - acc: 0.9430 - val_loss: 0.74
15 - val_acc: 0.9094
Epoch 28/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.4446 - acc: 0.9339 - val_loss: 0.80
32 - val_acc: 0.9087
Epoch 29/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.3447 - acc: 0.9286 - val_loss: 0.84
79 - val_acc: 0.9036
Epoch 30/30
7352/7352 [==============================] - 30s 4ms/step - loss: 0.6636 - acc: 0.9180 - val_loss: 0.75
84 - val_acc: 0.8999
Test loss: 0.7584496770992061
Test accuracy: 0.8998982015609094
```

In [91]:

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

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
```

```
True
LAYING                  511         0         0         0                   0
SITTING                   1       372       111         4                   0
STANDING                  0        75       453         3                   0
WALKING                   0         0         0       492                   2
WALKING_DOWNSTAIRS        0         0         0        15                 404
WALKING_UPSTAIRS          0         0         0        42                   9

Pred                WALKING_UPSTAIRS
True
LAYING                            26
SITTING                            3
STANDING                           1
WALKING                            2
WALKING_DOWNSTAIRS                 1
WALKING_UPSTAIRS                 420
```

In [92]:

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

```
2947/2947 [==============================] - 3s 865us/step
```

Out[92]:

```
[0.7584496770992061, 0.8998982015609094]
```

In [93]:

```
out_table.add_row(["CNN","64-64-128 kernal=5,7,7","50%",score])
print(out_table)
```

```
+-----------------------------------------+-----------------------+-----------+-----------------------
----------------------+
|                  MODEL                  |      Hidden unit      |  Dropout  |                  Accu
rcy                   |
+-----------------------------------------+-----------------------+-----------+-----------------------
----------------------+
|               LSTM One                  |          32           |    50%    |          [0.6969475907
087326, 0.6]          |
|               LSTM One                  |          64           |    50%    |         [1.08183042234
18236, 0.566]         |
|               LSTM One                  |          64           |    70%    |         [0.91488204717
63611, 0.804]         |
|               LSTM Two                  |          64           |    80%    |         [1.13715102674
43954, 0.578]         |
|          LSTM One - Highdata            |          64           |    80%    |           [nan, 0.1683
06752629793]          |
|        LSTM One - Highdata + Relu       |          64           |    70%    |           [nan, 0.1683
06752629793]          |
|      LSTM One - Highdata + sigmoid      |          64           |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |          64           |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|          LSTM Two - Highdata            |          32           |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|          LSTM Two - Highdata            |        128 - 64       | 50% - 70% |           [nan, 0.1683
06752629793]          |
|          LSTM Two - Highdata            |        128 - 64       | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                CNN                      |        32 - 64        |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                CNN                      |        64 - 128       |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                CNN                      |   64 - 128 kernal=20  |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                CNN                      | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                CNN                      | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
+-----------------------------------------+-----------------------+-----------+-----------------------
----------------------+
```

Nothing is helping on improving the performance, we will increae the epoch on the model (with some modification) for which we have received the highest accuracy.

We will try to change the relu activation to sigmoid and increase the dropout rate also

CNN Model6

In [94]:

```
model = Sequential()
model.add(Conv1D(64, kernel_size=5,
                 activation='sigmoid',
                 input_shape=(timesteps, input_dim)))
model.add(Conv1D(128, 7, activation='sigmoid'))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='sigmoid'))
model.add(Dropout(0.7))
model.add(Dense(n_classes, activation='softmax'))

model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_30 (Conv1D)           (None, 124, 64)           2944
_________________________________________________________________
conv1d_31 (Conv1D)           (None, 118, 128)          57472
_________________________________________________________________
max_pooling1d_14 (MaxPooling (None, 29, 128)           0
_________________________________________________________________
flatten_11 (Flatten)         (None, 3712)              0
_________________________________________________________________
dense_21 (Dense)             (None, 128)               475264
_________________________________________________________________
dropout_16 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_22 (Dense)             (None, 6)                 774
=================================================================
Total params: 536,454
Trainable params: 536,454
Non-trainable params: 0
_________________________________________________________________
```

In [95]:

```
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 26s 3ms/step - loss: 1.9346 - acc: 0.1790 - val_loss: 1.79
01 - val_acc: 0.1666
Epoch 2/30
7352/7352 [==============================] - 25s 3ms/step - loss: 1.8070 - acc: 0.1828 - val_loss: 1.79
15 - val_acc: 0.1822
Epoch 3/30
7352/7352 [==============================] - 25s 3ms/step - loss: 1.7915 - acc: 0.1862 - val_loss: 1.78
86 - val_acc: 0.1822
Epoch 4/30
7352/7352 [==============================] - 25s 3ms/step - loss: 1.7973 - acc: 0.1821 - val_loss: 1.79
```

```
03 - val_acc: 0.1805
Epoch 5/30
7352/7352 [==============================] - 24s 3ms/step - loss: 1.7993 - acc: 0.1819 - val_loss: 1.79
22 - val_acc: 0.1822
Epoch 6/30
7352/7352 [==============================] - 25s 3ms/step - loss: 1.7936 - acc: 0.1884 - val_loss: 1.79
33 - val_acc: 0.1822
Epoch 7/30
7352/7352 [==============================] - 25s 3ms/step - loss: 1.7928 - acc: 0.1854 - val_loss: 1.79
36 - val_acc: 0.1822
Epoch 8/30
7352/7352 [==============================] - 25s 3ms/step - loss: 1.7928 - acc: 0.1870 - val_loss: 1.79
11 - val_acc: 0.1822
Epoch 9/30
7352/7352 [==============================] - 26s 4ms/step - loss: 1.7922 - acc: 0.1801 - val_loss: 1.79
02 - val_acc: 0.1822
Epoch 10/30
7352/7352 [==============================] - 26s 3ms/step - loss: 1.7917 - acc: 0.1868 - val_loss: 1.79
14 - val_acc: 0.1666
Epoch 11/30
7352/7352 [==============================] - 26s 4ms/step - loss: 1.7915 - acc: 0.1821 - val_loss: 1.79
46 - val_acc: 0.1805
Epoch 12/30
7352/7352 [==============================] - 26s 4ms/step - loss: 1.7943 - acc: 0.1821 - val_loss: 1.78
92 - val_acc: 0.1822
Epoch 13/30
7352/7352 [==============================] - 24s 3ms/step - loss: 1.7921 - acc: 0.1776 - val_loss: 1.78
93 - val_acc: 0.1683
Epoch 14/30
7352/7352 [==============================] - 24s 3ms/step - loss: 1.7908 - acc: 0.1888 - val_loss: 1.79
13 - val_acc: 0.1822
Epoch 15/30
7352/7352 [==============================] - 25s 3ms/step - loss: 1.7913 - acc: 0.1821 - val_loss: 1.79
05 - val_acc: 0.1822
Epoch 16/30
7352/7352 [==============================] - 25s 3ms/step - loss: 1.7910 - acc: 0.1861 - val_loss: 1.78
97 - val_acc: 0.1822
Epoch 17/30
7352/7352 [==============================] - 26s 4ms/step - loss: 1.7906 - acc: 0.1854 - val_loss: 1.79
16 - val_acc: 0.1822
Epoch 18/30
7352/7352 [==============================] - 26s 4ms/step - loss: 1.7913 - acc: 0.1861 - val_loss: 1.79
17 - val_acc: 0.1805
Epoch 19/30
7352/7352 [==============================] - 27s 4ms/step - loss: 1.7898 - acc: 0.1866 - val_loss: 1.79
38 - val_acc: 0.1805
Epoch 20/30
7352/7352 [==============================] - 27s 4ms/step - loss: 1.7900 - acc: 0.1802 - val_loss: 1.79
34 - val_acc: 0.1822
Epoch 21/30
7352/7352 [==============================] - 26s 4ms/step - loss: 1.7910 - acc: 0.1876 - val_loss: 1.78
90 - val_acc: 0.1805
Epoch 22/30
7352/7352 [==============================] - 26s 4ms/step - loss: 1.7921 - acc: 0.1802 - val_loss: 1.79
00 - val_acc: 0.1822
Epoch 23/30
7352/7352 [==============================] - 26s 3ms/step - loss: 1.7911 - acc: 0.1802 - val_loss: 1.79
28 - val_acc: 0.1666
Epoch 24/30
7352/7352 [==============================] - 25s 3ms/step - loss: 1.7918 - acc: 0.1854 - val_loss: 1.79
17 - val_acc: 0.1805
Epoch 25/30
7352/7352 [==============================] - 24s 3ms/step - loss: 1.7925 - acc: 0.1810 - val_loss: 1.79
07 - val_acc: 0.1822
Epoch 26/30
7352/7352 [==============================] - 24s 3ms/step - loss: 1.7907 - acc: 0.1813 - val_loss: 1.79
04 - val_acc: 0.1822
Epoch 27/30
7352/7352 [==============================] - 25s 3ms/step - loss: 1.7918 - acc: 0.1828 - val_loss: 1.79
32 - val_acc: 0.1805
Epoch 28/30
7352/7352 [==============================] - 24s 3ms/step - loss: 1.7919 - acc: 0.1855 - val_loss: 1.79
30 - val_acc: 0.1822
Epoch 29/30
7352/7352 [==============================] - 25s 3ms/step - loss: 1.7913 - acc: 0.1793 - val_loss: 1.79
05 - val_acc: 0.1805
Epoch 30/30
```

```
Epoch 30/30
7352/7352 [==============================] - 24s 3ms/step - loss: 1.7910 - acc: 0.1834 - val_loss: 1.79
38 - val_acc: 0.1822
Test loss: 1.793824812902367
Test accuracy: 0.18221920597217509
```

In [96]:

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

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

out_table.add_row(["CNN - sigmoid","64-128 kernal=5,7","70%",score])
print(out_table)
```

```
Pred                LAYING
True
LAYING                 537
SITTING                491
STANDING               532
WALKING                496
WALKING_DOWNSTAIRS     420
WALKING_UPSTAIRS       471
2947/2947 [==============================] - 2s 679us/step
[1.793824812902367, 0.18221920597217509]
+-----------------------------------------+----------------------+-----------+----------------------
---------------------+
|                  MODEL                  |      Hidden unit     |  Dropout  |                  Accu
rcy                  |
+-----------------------------------------+----------------------+-----------+----------------------
---------------------+
|                LSTM One                 |          32          |    50%    |          [0.6969475907
087326, 0.6]         |
|                LSTM One                 |          64          |    50%    |         [1.08183042234
18236, 0.566]        |
|                LSTM One                 |          64          |    70%    |         [0.91488204717
63611, 0.804]        |
|                LSTM Two                 |          64          |    80%    |         [1.13715102674
43954, 0.578]        |
|           LSTM One - Highdata           |          64          |    80%    |           [nan, 0.1683
06752629793]         |
|        LSTM One - Highdata + Relu       |          64          |    70%    |           [nan, 0.1683
06752629793]         |
|      LSTM One - Highdata + sigmoid      |          64          |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |          64          |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|           LSTM Two - Highdata           |          32          |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|           LSTM Two - Highdata           |       128 - 64       | 50% - 70% |           [nan, 0.1683
06752629793]         |
|           LSTM Two - Highdata           |       128 - 64       | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                  CNN                    |       32 - 64        |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                  CNN                    |       64 - 128       |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                  CNN                    |   64 - 128 kernal=20 |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                  CNN                    | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                  CNN                    | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
|              CNN - sigmoid              |   64-128 kernal=5,7  |    70%    |  [1.793824812902367,
0.18221920597217509] |
+-----------------------------------------+----------------------+-----------+----------------------
---------------------+
```

Very bad result with sigmoid, we will try the similar with relu

CNN Model 7

CNN Model

In [97]:

```
model = Sequential()
model.add(Conv1D(64, kernel_size=5,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(Conv1D(128, 7, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.7))
model.add(Dense(n_classes, activation='softmax'))

model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_32 (Conv1D)           (None, 124, 64)           2944
_________________________________________________________________
conv1d_33 (Conv1D)           (None, 118, 128)          57472
_________________________________________________________________
max_pooling1d_15 (MaxPooling (None, 29, 128)           0
_________________________________________________________________
flatten_12 (Flatten)         (None, 3712)              0
_________________________________________________________________
dense_23 (Dense)             (None, 128)               475264
_________________________________________________________________
dropout_17 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_24 (Dense)             (None, 6)                 774
=================================================================
Total params: 536,454
Trainable params: 536,454
Non-trainable params: 0
_________________________________________________________________
```

In [98]:

```
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.6425 - acc: 0.7557 - val_loss: 0.40
20 - val_acc: 0.8962
Epoch 2/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.2282 - acc: 0.9089 - val_loss: 0.35
27 - val_acc: 0.8985
Epoch 3/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.2062 - acc: 0.9136 - val_loss: 0.37
76 - val_acc: 0.9026
Epoch 4/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.1791 - acc: 0.9272 - val_loss: 0.42
33 - val_acc: 0.9030
Epoch 5/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.1648 - acc: 0.9320 - val_loss: 0.47
52 - val_acc: 0.9135
Epoch 6/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.1557 - acc: 0.9362 - val_loss: 0.21
27 - val_acc: 0.9216
Epoch 7/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.1531 - acc: 0.9378 - val loss: 0.27
```

```
18 - val_acc: 0.9141
Epoch 8/30
7352/7352 [==============================] - 26s 3ms/step - loss: 0.1438 - acc: 0.9400 - val_loss: 0.34
93 - val_acc: 0.9033
Epoch 9/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.1440 - acc: 0.9419 - val_loss: 0.39
55 - val_acc: 0.9108
Epoch 10/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.1520 - acc: 0.9384 - val_loss: 0.45
94 - val_acc: 0.9128
Epoch 11/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.1658 - acc: 0.9418 - val_loss: 0.45
76 - val_acc: 0.9050
Epoch 12/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.1376 - acc: 0.9429 - val_loss: 0.29
13 - val_acc: 0.9182
Epoch 13/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.1294 - acc: 0.9475 - val_loss: 0.26
16 - val_acc: 0.9203
Epoch 14/30
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1302 - acc: 0.9431 - val_loss: 0.27
40 - val_acc: 0.9237
Epoch 15/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1191 - acc: 0.9457 - val_loss: 0.32
48 - val_acc: 0.9165
Epoch 16/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.1276 - acc: 0.9497 - val_loss: 0.33
40 - val_acc: 0.9162
Epoch 17/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1182 - acc: 0.9491 - val_loss: 0.40
07 - val_acc: 0.9158
Epoch 18/30
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1122 - acc: 0.9520 - val_loss: 0.43
07 - val_acc: 0.9131
Epoch 19/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1130 - acc: 0.9514 - val_loss: 0.46
84 - val_acc: 0.9179
Epoch 20/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1087 - acc: 0.9528 - val_loss: 0.52
01 - val_acc: 0.9165
Epoch 21/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.0979 - acc: 0.9543 - val_loss: 0.33
07 - val_acc: 0.9325
Epoch 22/30
7352/7352 [==============================] - 30s 4ms/step - loss: 0.0933 - acc: 0.9557 - val_loss: 0.41
85 - val_acc: 0.9284
Epoch 23/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.0869 - acc: 0.9567 - val_loss: 0.61
45 - val_acc: 0.9138
Epoch 24/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.1515 - acc: 0.9512 - val_loss: 0.61
19 - val_acc: 0.9145
Epoch 25/30
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1053 - acc: 0.9524 - val_loss: 0.73
49 - val_acc: 0.9097
Epoch 26/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.0849 - acc: 0.9610 - val_loss: 0.58
22 - val_acc: 0.9118
Epoch 27/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.0780 - acc: 0.9621 - val_loss: 0.58
70 - val_acc: 0.9206
Epoch 28/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.0992 - acc: 0.9578 - val_loss: 0.67
90 - val_acc: 0.9145
Epoch 29/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.2132 - acc: 0.9533 - val_loss: 0.60
54 - val_acc: 0.9175
Epoch 30/30
7352/7352 [==============================] - 26s 3ms/step - loss: 0.1143 - acc: 0.9601 - val_loss: 0.44
98 - val_acc: 0.9216
Test loss: 0.4498406055611254
Test accuracy: 0.9216152019002375
```

In [101]:

```
# Confusion Matrix
```

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

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

out_table.add_row(["CNN - relu","64-128 kernal=5,7","70%",score])
print(out_table)
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 536        0         0        0                   0
SITTING                  4      368       113        0                   0
STANDING                 0       71       461        0                   0
WALKING                  0        0         0      477                   2
WALKING_DOWNSTAIRS       0        0         0        0                 418
WALKING_UPSTAIRS         0        0         0        3                  12

Pred                WALKING_UPSTAIRS
True
LAYING                             1
SITTING                            6
STANDING                           0
WALKING                           17
WALKING_DOWNSTAIRS                 2
WALKING_UPSTAIRS                 456
2947/2947 [==============================] - 2s 689us/step
[0.4498406055611254, 0.9216152019002375]
+-----------------------------------------+------------------------+-----------+----------------------
---------------------+
|                MODEL                    |      Hidden unit       |  Dropout  |                   Accu
rcy                  |
+-----------------------------------------+------------------------+-----------+----------------------
---------------------+
|               LSTM One                  |           32           |    50%    |          [0.6969475907
087326, 0.6]         |
|               LSTM One                  |           64           |    50%    |         [1.08183042234
18236, 0.566]        |
|               LSTM One                  |           64           |    70%    |         [0.91488204717
63611, 0.804]        |
|               LSTM Two                  |           64           |    80%    |         [1.13715102674
43954, 0.578]        |
|          LSTM One - Highdata            |           64           |    80%    |           [nan, 0.1683
06752629793]         |
|       LSTM One - Highdata + Relu        |           64           |    70%    |           [nan, 0.1683
06752629793]         |
|     LSTM One - Highdata + sigmoid       |           64           |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |           64           |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|          LSTM Two - Highdata            |           32           |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|          LSTM Two - Highdata            |        128 - 64        | 50% - 70% |           [nan, 0.1683
06752629793]         |
|          LSTM Two - Highdata            |        128 - 64        | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                CNN                      |        32 - 64         |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                CNN                      |        64 - 128        |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                CNN                      |   64 - 128 kernal=20   |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                CNN                      | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                CNN                      | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
|            CNN - sigmoid                |   64-128 kernal=5,7    |    70%    |  [1.793824812902367,
0.18221920597217509] |
|              CNN - relu                 |   64-128 kernal=5,7    |    70%    |  [0.4498406055611254,
0.9216152019002375] |
+-----------------------------------------+------------------------+-----------+----------------------
---------------------+
```

CNN Model7

In [105]:

```
model = Sequential()
model.add(Conv1D(64, kernel_size=50,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(Conv1D(128, 5, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.7))
model.add(Dense(n_classes, activation='softmax'))

model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_40 (Conv1D)           (None, 79, 64)            28864
_________________________________________________________________
conv1d_41 (Conv1D)           (None, 75, 128)           41088
_________________________________________________________________
max_pooling1d_16 (MaxPooling (None, 18, 128)           0
_________________________________________________________________
flatten_13 (Flatten)         (None, 2304)              0
_________________________________________________________________
dense_25 (Dense)             (None, 128)               295040
_________________________________________________________________
dropout_18 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_26 (Dense)             (None, 6)                 774
=================================================================
Total params: 365,766
Trainable params: 365,766
Non-trainable params: 0
_________________________________________________________________
```

In [106]:

```
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 20s 3ms/step - loss: 0.5258 - acc: 0.7962 - val_loss: 0.25
46 - val_acc: 0.9060
Epoch 2/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.2055 - acc: 0.9253 - val_loss: 0.36
50 - val_acc: 0.9043
Epoch 3/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1700 - acc: 0.9408 - val_loss: 0.33
36 - val_acc: 0.9060
Epoch 4/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1693 - acc: 0.9353 - val_loss: 0.39
20 - val_acc: 0.9101
Epoch 5/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1355 - acc: 0.9475 - val_loss: 0.81
32 - val_acc: 0.8599
Epoch 6/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1428 - acc: 0.9427 - val_loss: 0.58
84 - val_acc: 0.9077
Epoch 7/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.2565 - acc: 0.9342 - val_loss: 0.51
78 - val_acc: 0.9165
```

```
Epoch 8/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.2079 - acc: 0.9397 - val_loss: 0.50
78 - val_acc: 0.9091
Epoch 9/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1804 - acc: 0.9445 - val_loss: 0.50
20 - val_acc: 0.9175
Epoch 10/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1585 - acc: 0.9474 - val_loss: 0.55
68 - val_acc: 0.9192
Epoch 11/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.2399 - acc: 0.9382 - val_loss: 0.52
90 - val_acc: 0.9141
Epoch 12/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1663 - acc: 0.9465 - val_loss: 0.76
92 - val_acc: 0.9087
Epoch 13/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1706 - acc: 0.9472 - val_loss: 0.72
50 - val_acc: 0.8975
Epoch 14/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1764 - acc: 0.9465 - val_loss: 0.76
18 - val_acc: 0.8999
Epoch 15/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1745 - acc: 0.9464 - val_loss: 0.72
88 - val_acc: 0.9101
Epoch 16/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1337 - acc: 0.9528 - val_loss: 0.74
30 - val_acc: 0.9019
Epoch 17/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1358 - acc: 0.9540 - val_loss: 0.74
26 - val_acc: 0.9135
Epoch 18/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1308 - acc: 0.9553 - val_loss: 0.82
80 - val_acc: 0.9108
Epoch 19/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1697 - acc: 0.9528 - val_loss: 0.88
52 - val_acc: 0.9050
Epoch 20/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1560 - acc: 0.9551 - val_loss: 0.81
43 - val_acc: 0.9040
Epoch 21/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1168 - acc: 0.9536 - val_loss: 0.80
89 - val_acc: 0.9026
Epoch 22/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1233 - acc: 0.9518 - val_loss: 0.85
47 - val_acc: 0.8972
Epoch 23/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1131 - acc: 0.9547 - val_loss: 0.81
70 - val_acc: 0.8979
Epoch 24/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1227 - acc: 0.9555 - val_loss: 0.84
02 - val_acc: 0.9077
Epoch 25/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1675 - acc: 0.9497 - val_loss: 0.94
67 - val_acc: 0.8941
Epoch 26/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.2242 - acc: 0.9475 - val_loss: 0.60
81 - val_acc: 0.9169
Epoch 27/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.2150 - acc: 0.9501 - val_loss: 0.68
27 - val_acc: 0.9152
Epoch 28/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.2127 - acc: 0.9504 - val_loss: 0.70
93 - val_acc: 0.9063
Epoch 29/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1627 - acc: 0.9531 - val_loss: 0.80
05 - val_acc: 0.9111
Epoch 30/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1887 - acc: 0.9520 - val_loss: 0.72
04 - val_acc: 0.9179
Test loss: 0.7203596894995468
Test accuracy: 0.9178825924669155
```

In [107]:

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

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

out_table.add_row(["CNN - relu","64-128 kernal=50,5","70%",score])
print(out_table)
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 537        0         0        0                   0
SITTING                  6      403        80        0                   0
STANDING                 0      102       429        1                   0
WALKING                  0        0         0      468                  27
WALKING_DOWNSTAIRS       0        0         0        0                 420
WALKING_UPSTAIRS         0        0         0        0                  23

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            2
STANDING                           0
WALKING                            1
WALKING_DOWNSTAIRS                 0
WALKING_UPSTAIRS                 448
2947/2947 [==============================] - 2s 717us/step
[0.7203596894995468, 0.9178825924669155]
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                  MODEL                  |      Hidden unit      |  Dropout  |                   Accu
rcy                   |
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|               LSTM One                  |           32          |    50%    |          [0.6969475907
087326, 0.6]          |
|               LSTM One                  |           64          |    50%    |         [1.08183042234
18236, 0.566]         |
|               LSTM One                  |           64          |    70%    |         [0.91488204717
63611, 0.804]         |
|               LSTM Two                  |           64          |    80%    |         [1.13715102674
43954, 0.578]         |
|           LSTM One - Highdata           |           64          |    80%    |           [nan, 0.1683
06752629793]          |
|        LSTM One - Highdata + Relu       |           64          |    70%    |           [nan, 0.1683
06752629793]          |
|      LSTM One - Highdata + sigmoid      |           64          |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |           64          |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|           LSTM Two - Highdata           |           32          |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|           LSTM Two - Highdata           |        128 - 64       | 50% - 70% |           [nan, 0.1683
06752629793]          |
|           LSTM Two - Highdata           |        128 - 64       | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                  CNN                    |        32 - 64        |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                  CNN                    |        64 - 128       |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                  CNN                    |   64 - 128 kernal=20  |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                  CNN                    | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                  CNN                    | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
|              CNN - sigmoid              |   64-128 kernal=5,7   |    70%    |  [1.793824812902367,
0.18221920597217509] |
|                CNN - relu               |   64-128 kernal=5,7   |    70%    |  [0.4498406055611254,
0.9216152019002375] |
|                CNN - relu               |   64-128 kernal=50,5  |    70%    |  [0.7203596894995468,
0.9178825924669155] |
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
```

CNN Model 8

CNN Model 3

In [108]:

```
model = Sequential()
model.add(Conv1D(128, kernel_size=7,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(Conv1D(64, 5, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.8))
model.add(Dense(n_classes, activation='softmax'))

model.summary()



model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_42 (Conv1D)           (None, 122, 128)          8192
_________________________________________________________________
conv1d_43 (Conv1D)           (None, 118, 64)           41024
_________________________________________________________________
max_pooling1d_17 (MaxPooling (None, 29, 64)            0
_________________________________________________________________
flatten_14 (Flatten)         (None, 1856)              0
_________________________________________________________________
dense_27 (Dense)             (None, 128)               237696
_________________________________________________________________
dropout_19 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_28 (Dense)             (None, 6)                 774
=================================================================
Total params: 287,686
Trainable params: 287,686
Non-trainable params: 0
_________________________________________________________________
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 20s 3ms/step - loss: 0.7560 - acc: 0.7005 - val_loss: 0.35
20 - val_acc: 0.8568
Epoch 2/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.2907 - acc: 0.8815 - val_loss: 0.25
33 - val_acc: 0.9057
Epoch 3/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.2335 - acc: 0.9057 - val_loss: 0.47
51 - val_acc: 0.9169
Epoch 4/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1964 - acc: 0.9178 - val_loss: 0.24
07 - val_acc: 0.9257
Epoch 5/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1778 - acc: 0.9204 - val_loss: 0.22
73 - val_acc: 0.9175
Epoch 6/30
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1784 - acc: 0.9285 - val_loss: 0.23
51 - val_acc: 0.9318
Epoch 7/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1569 - acc: 0.9323 - val_loss: 0.47
99 - val_acc: 0.8982
Epoch 8/30
```

```
Epoch 8/30
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1736 - acc: 0.9304 - val_loss: 0.41
03 - val_acc: 0.9152
Epoch 9/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1521 - acc: 0.9354 - val_loss: 0.45
34 - val_acc: 0.9087
Epoch 10/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1348 - acc: 0.9406 - val_loss: 0.48
35 - val_acc: 0.9199
Epoch 11/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1236 - acc: 0.9438 - val_loss: 0.49
98 - val_acc: 0.9203
Epoch 12/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1937 - acc: 0.9400 - val_loss: 0.49
06 - val_acc: 0.9250
Epoch 13/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1263 - acc: 0.9453 - val_loss: 0.52
65 - val_acc: 0.9250
Epoch 14/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1125 - acc: 0.9498 - val_loss: 0.55
54 - val_acc: 0.9189
Epoch 15/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1046 - acc: 0.9514 - val_loss: 0.57
74 - val_acc: 0.9084
Epoch 16/30
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1025 - acc: 0.9513 - val_loss: 0.92
29 - val_acc: 0.8924
Epoch 17/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1539 - acc: 0.9463 - val_loss: 0.63
27 - val_acc: 0.9158
Epoch 18/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1070 - acc: 0.9517 - val_loss: 0.60
10 - val_acc: 0.9264
Epoch 19/30
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1386 - acc: 0.9538 - val_loss: 0.50
10 - val_acc: 0.9355
Epoch 20/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1443 - acc: 0.9497 - val_loss: 0.40
95 - val_acc: 0.9240
Epoch 21/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1150 - acc: 0.9540 - val_loss: 0.57
18 - val_acc: 0.9270
Epoch 22/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1052 - acc: 0.9527 - val_loss: 0.59
42 - val_acc: 0.9179
Epoch 23/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.0988 - acc: 0.9544 - val_loss: 0.46
77 - val_acc: 0.9389
Epoch 24/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.0858 - acc: 0.9603 - val_loss: 0.41
83 - val_acc: 0.9352
Epoch 25/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.0925 - acc: 0.9587 - val_loss: 0.54
06 - val_acc: 0.9077
Epoch 26/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1233 - acc: 0.9553 - val_loss: 0.65
47 - val_acc: 0.9233
Epoch 27/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.0995 - acc: 0.9591 - val_loss: 0.58
75 - val_acc: 0.9315
Epoch 28/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.0940 - acc: 0.9596 - val_loss: 0.66
46 - val_acc: 0.9230
Epoch 29/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.0893 - acc: 0.9607 - val_loss: 0.68
07 - val_acc: 0.9243
Epoch 30/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.0994 - acc: 0.9616 - val_loss: 0.67
93 - val_acc: 0.9226
Test loss: 0.6793473407663981
Test accuracy: 0.9226331862911435
```

In [109]:

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

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

out_table.add_row(["CNN - relu","128-64 kernal=50,5","80%",score])
print(out_table)
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 511        0         0        0                   0
SITTING                  5      393        88        0                   0
STANDING                 0       66       465        1                   0
WALKING                  0        0         0      471                  24
WALKING_DOWNSTAIRS       0        0         0        1                 419
WALKING_UPSTAIRS         0        0         0        0                  11

Pred                WALKING_UPSTAIRS
True
LAYING                            26
SITTING                            5
STANDING                           0
WALKING                            1
WALKING_DOWNSTAIRS                 0
WALKING_UPSTAIRS                 460
2947/2947 [==============================] - 2s 631us/step
[0.6793473407663981, 0.9226331862911435]
+-----------------------------------------+----------------------+-----------+----------------------
---------------------+
|                  MODEL                  |      Hidden unit     |  Dropout  |                   Accu
rcy                   |
+-----------------------------------------+----------------------+-----------+----------------------
---------------------+
|                LSTM One                 |          32          |    50%    |          [0.6969475907
087326, 0.6]          |
|                LSTM One                 |          64          |    50%    |         [1.08183042234
18236, 0.566]         |
|                LSTM One                 |          64          |    70%    |         [0.91488204717
63611, 0.804]         |
|                LSTM Two                 |          64          |    80%    |         [1.13715102674
43954, 0.578]         |
|           LSTM One - Highdata           |          64          |    80%    |           [nan, 0.1683
06752629793]          |
|        LSTM One - Highdata + Relu       |          64          |    70%    |           [nan, 0.1683
06752629793]          |
|      LSTM One - Highdata + sigmoid      |          64          |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |          64          |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|           LSTM Two - Highdata           |          32          |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|           LSTM Two - Highdata           |       128 - 64       | 50% - 70% |           [nan, 0.1683
06752629793]          |
|           LSTM Two - Highdata           |       128 - 64       | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                CNN                      |       32 - 64        |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                CNN                      |       64 - 128       |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                CNN                      |   64 - 128 kernal=20   |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                CNN                      | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                CNN                      | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
|             CNN - sigmoid               |   64-128 kernal=5,7    |    70%    |  [1.793824812902367,
0.18221920597217509] |
|               CNN - relu                |   64-128 kernal=5,7    |    70%    |  [0.4498406055611254,
0.9216152019002375] |
|               CNN - relu                |   64-128 kernal=50,5   |    70%    |  [0.7203596894995468,
0.9178825924669155] |
|               CNN - relu                |   128-64 kernal=50,5   |    80%    |  [0.6793473407663981,
0.9226331862911435] |
+-----------------------------------------+----------------------+-----------+----------------------
---------------------+
```

Certain improvement in training can be observed. Lets decrease the filters further more to observe accuracy

CNN Model8

In [110]:

```
model = Sequential()
model.add(Conv1D(64, kernel_size=5,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(Conv1D(32, 3, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.8))
model.add(Dense(n_classes, activation='softmax'))

model.summary()




model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_44 (Conv1D)           (None, 124, 64)           2944
_________________________________________________________________
conv1d_45 (Conv1D)           (None, 122, 32)           6176
_________________________________________________________________
max_pooling1d_18 (MaxPooling (None, 30, 32)            0
_________________________________________________________________
flatten_15 (Flatten)         (None, 960)               0
_________________________________________________________________
dense_29 (Dense)             (None, 128)               123008
_________________________________________________________________
dropout_20 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_30 (Dense)             (None, 6)                 774
=================================================================
Total params: 132,902
Trainable params: 132,902
Non-trainable params: 0
_________________________________________________________________
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 9s 1ms/step - loss: 0.7722 - acc: 0.6678 - val_loss: 0.425
8 - val_acc: 0.8565
Epoch 2/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.3726 - acc: 0.8500 - val_loss: 0.393
1 - val_acc: 0.8721
Epoch 3/30
7352/7352 [==============================] - 7s 976us/step - loss: 0.2668 - acc: 0.8908 - val_loss: 0.3
604 - val_acc: 0.8931
Epoch 4/30
7352/7352 [==============================] - 7s 1ms/step - loss: 0.2433 - acc: 0.8998 - val_loss: 0.380
4 - val_acc: 0.8870
Epoch 5/30
7352/7352 [==============================] - 7s 995us/step - loss: 0.1988 - acc: 0.9177 - val_loss: 0.3
058 - val_acc: 0.9019
Epoch 6/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1861 - acc: 0.9248 - val_loss: 0.343
8 - val_acc: 0.8962
```

```
6 - val_acc: 0.8962
Epoch 7/30
7352/7352 [==============================] - 7s 941us/step - loss: 0.1637 - acc: 0.9238 - val_loss: 0.3
588 - val_acc: 0.8945
Epoch 8/30
7352/7352 [==============================] - 7s 991us/step - loss: 0.1807 - acc: 0.9233 - val_loss: 0.3
955 - val_acc: 0.8951
Epoch 9/30
7352/7352 [==============================] - 7s 1ms/step - loss: 0.1771 - acc: 0.9294 - val_loss: 0.348
4 - val_acc: 0.9114
Epoch 10/30
7352/7352 [==============================] - 7s 979us/step - loss: 0.1514 - acc: 0.9365 - val_loss: 0.4
030 - val_acc: 0.9108
Epoch 11/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1491 - acc: 0.9346 - val_loss: 0.305
9 - val_acc: 0.9230
Epoch 12/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1402 - acc: 0.9346 - val_loss: 0.325
3 - val_acc: 0.9182
Epoch 13/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1452 - acc: 0.9381 - val_loss: 0.319
4 - val_acc: 0.9155
Epoch 14/30
7352/7352 [==============================] - 7s 995us/step - loss: 0.1327 - acc: 0.9399 - val_loss: 0.3
664 - val_acc: 0.9074
Epoch 15/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1423 - acc: 0.9362 - val_loss: 0.528
2 - val_acc: 0.9067
Epoch 16/30
7352/7352 [==============================] - 7s 996us/step - loss: 0.1225 - acc: 0.9445 - val_loss: 0.5
183 - val_acc: 0.9046
Epoch 17/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1397 - acc: 0.9400 - val_loss: 0.403
6 - val_acc: 0.9101
Epoch 18/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1175 - acc: 0.9436 - val_loss: 0.369
4 - val_acc: 0.9203
Epoch 19/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1182 - acc: 0.9448 - val_loss: 0.703
4 - val_acc: 0.8877
Epoch 20/30
7352/7352 [==============================] - 7s 953us/step - loss: 0.1197 - acc: 0.9459 - val_loss: 0.6
014 - val_acc: 0.9165
Epoch 21/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1133 - acc: 0.9478 - val_loss: 0.525
8 - val_acc: 0.9125
Epoch 22/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1143 - acc: 0.9483 - val_loss: 0.598
0 - val_acc: 0.9077
Epoch 23/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1190 - acc: 0.9467 - val_loss: 0.693
5 - val_acc: 0.9097
Epoch 24/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1117 - acc: 0.9484 - val_loss: 0.605
4 - val_acc: 0.9213
Epoch 25/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.0998 - acc: 0.9540 - val_loss: 0.600
4 - val_acc: 0.9172
Epoch 26/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1055 - acc: 0.9543 - val_loss: 0.700
3 - val_acc: 0.9016
Epoch 27/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.0945 - acc: 0.9529 - val_loss: 0.653
2 - val_acc: 0.9070
Epoch 28/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1034 - acc: 0.9540 - val_loss: 0.659
9 - val_acc: 0.9108
Epoch 29/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1039 - acc: 0.9512 - val_loss: 0.578
3 - val_acc: 0.9084
Epoch 30/30
7352/7352 [==============================] - 8s 1ms/step - loss: 0.1069 - acc: 0.9527 - val_loss: 0.743
1 - val_acc: 0.9057
Test loss: 0.7431288106871484
Test accuracy: 0.9056667797760435
```

In [114]:

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

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

out_table.add_row(["CNN - relu","64-32 kernal=5,3","80%",score])
print(out_table)
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 510        0         0        0                   0
SITTING                  0      367       114        1                   0
STANDING                 0       59       471        0                   0
WALKING                  0        0         0      461                  34
WALKING_DOWNSTAIRS       0        0         0        0                 420
WALKING_UPSTAIRS         0        0         0        0                  31

Pred                WALKING_UPSTAIRS
True
LAYING                            27
SITTING                            9
STANDING                           2
WALKING                            1
WALKING_DOWNSTAIRS                 0
WALKING_UPSTAIRS                 440
2947/2947 [==============================] - 1s 277us/step
[0.7431288106871484, 0.9056667797760435]
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                  MODEL                  |      Hidden unit      |  Dropout  |                   Accu
rcy                   |
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                LSTM One                 |          32           |    50%    |          [0.6969475907
087326, 0.6]          |
|                LSTM One                 |          64           |    50%    |         [1.08183042234
18236, 0.566]         |
|                LSTM One                 |          64           |    70%    |         [0.91488204717
63611, 0.804]         |
|                LSTM Two                 |          64           |    80%    |         [1.13715102674
43954, 0.578]         |
|           LSTM One - Highdata           |          64           |    80%    |           [nan, 0.1683
06752629793]          |
|        LSTM One - Highdata + Relu       |          64           |    70%    |           [nan, 0.1683
06752629793]          |
|      LSTM One - Highdata + sigmoid      |          64           |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |          64           |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|           LSTM Two - Highdata           |          32           |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|           LSTM Two - Highdata           |        128 - 64       | 50% - 70% |           [nan, 0.1683
06752629793]          |
|           LSTM Two - Highdata           |        128 - 64       | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                CNN                      |        32 - 64        |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                CNN                      |        64 - 128       |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                CNN                      |   64 - 128 kernal=20  |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                CNN                      | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                CNN                      | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
|             CNN - sigmoid               |   64-128 kernal=5,7   |    70%    |  [1.793824812902367,
0.18221920597217509] |
|               CNN - relu                |   64-128 kernal=5,7   |    70%    |  [0.4498406055611254,
0.9216152019002375] |
|               CNN - relu                |   64-128 kernal=50,5  |    70%    |  [0.7203596894995468,
0.9178825924669155] |
|               CNN - relu                |   128-64 kernal=50,5  |    80%    |  [0.6793473407663981,
0.9226331862911435] |
|               CNN - relu                |    64-32 kernal=5,3   |    80%    |  [0.7431288106871484,
```

```
0.9056667797760435] |
+---------------------------------------+----------------------+----------+----------------------
--------------------+
```

Not seeing improvement as previous, Lets observe the performance with only One layer

CNN Model9

In [115]:

```
model = Sequential()
model.add(Conv1D(128, kernel_size=7,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.8))
model.add(Dense(n_classes, activation='softmax'))

model.summary()




model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_46 (Conv1D)           (None, 122, 128)          8192
_________________________________________________________________
max_pooling1d_19 (MaxPooling (None, 30, 128)           0
_________________________________________________________________
flatten_16 (Flatten)         (None, 3840)              0
_________________________________________________________________
dense_31 (Dense)             (None, 128)               491648
_________________________________________________________________
dropout_21 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_32 (Dense)             (None, 6)                 774
=================================================================
Total params: 500,614
Trainable params: 500,614
Non-trainable params: 0
_________________________________________________________________
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 13s 2ms/step - loss: 0.7149 - acc: 0.7130 - val_loss: 0.34
87 - val_acc: 0.8873
Epoch 2/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.3212 - acc: 0.8777 - val_loss: 0.24
72 - val_acc: 0.9036
Epoch 3/30
7352/7352 [==============================] - 11s 1ms/step - loss: 0.2476 - acc: 0.9004 - val_loss: 0.34
90 - val_acc: 0.9013
Epoch 4/30
7352/7352 [==============================] - 11s 1ms/step - loss: 0.2233 - acc: 0.9097 - val_loss: 0.20
02 - val_acc: 0.9199
Epoch 5/30
7352/7352 [==============================] - 11s 1ms/step - loss: 0.1832 - acc: 0.9230 - val_loss: 0.26
03 - val_acc: 0.9033
```

```
05 - val_acc: 0.9055
Epoch 6/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.2029 - acc: 0.9207 - val_loss: 0.25
88 - val_acc: 0.9111
Epoch 7/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1856 - acc: 0.9256 - val_loss: 0.26
90 - val_acc: 0.9162
Epoch 8/30
7352/7352 [==============================] - 11s 1ms/step - loss: 0.1736 - acc: 0.9314 - val_loss: 0.39
43 - val_acc: 0.9091
Epoch 9/30
7352/7352 [==============================] - 11s 2ms/step - loss: 0.1774 - acc: 0.9306 - val_loss: 0.20
20 - val_acc: 0.9287
Epoch 10/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1726 - acc: 0.9314 - val_loss: 0.19
39 - val_acc: 0.9243
Epoch 11/30
7352/7352 [==============================] - 11s 1ms/step - loss: 0.1679 - acc: 0.9313 - val_loss: 0.28
57 - val_acc: 0.9108
Epoch 12/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1509 - acc: 0.9407 - val_loss: 0.21
78 - val_acc: 0.9209
Epoch 13/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1461 - acc: 0.9372 - val_loss: 0.25
29 - val_acc: 0.9213
Epoch 14/30
7352/7352 [==============================] - 11s 2ms/step - loss: 0.1425 - acc: 0.9397 - val_loss: 0.27
26 - val_acc: 0.9162
Epoch 15/30
7352/7352 [==============================] - 13s 2ms/step - loss: 0.1444 - acc: 0.9412 - val_loss: 0.23
16 - val_acc: 0.9206
Epoch 16/30
7352/7352 [==============================] - 13s 2ms/step - loss: 0.1366 - acc: 0.9418 - val_loss: 0.23
82 - val_acc: 0.9243
Epoch 17/30
7352/7352 [==============================] - 12s 2ms/step - loss: 0.1400 - acc: 0.9450 - val_loss: 0.32
22 - val_acc: 0.9131
Epoch 18/30
7352/7352 [==============================] - 12s 2ms/step - loss: 0.1374 - acc: 0.9418 - val_loss: 0.26
09 - val_acc: 0.9267
Epoch 19/30
7352/7352 [==============================] - 12s 2ms/step - loss: 0.1406 - acc: 0.9392 - val_loss: 0.33
53 - val_acc: 0.9155
Epoch 20/30
7352/7352 [==============================] - 12s 2ms/step - loss: 0.1403 - acc: 0.9419 - val_loss: 0.39
78 - val_acc: 0.9230
Epoch 21/30
7352/7352 [==============================] - 11s 1ms/step - loss: 0.1321 - acc: 0.9406 - val_loss: 0.26
75 - val_acc: 0.9226
Epoch 22/30
7352/7352 [==============================] - 9s 1ms/step - loss: 0.1271 - acc: 0.9464 - val_loss: 0.401
1 - val_acc: 0.9101
Epoch 23/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1295 - acc: 0.9430 - val_loss: 0.38
87 - val_acc: 0.9138
Epoch 24/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1524 - acc: 0.9438 - val_loss: 0.23
10 - val_acc: 0.9318
Epoch 25/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1363 - acc: 0.9444 - val_loss: 0.24
29 - val_acc: 0.9287
Epoch 26/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1353 - acc: 0.9434 - val_loss: 0.24
10 - val_acc: 0.9267
Epoch 27/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1347 - acc: 0.9442 - val_loss: 0.25
79 - val_acc: 0.9230
Epoch 28/30
7352/7352 [==============================] - 12s 2ms/step - loss: 0.1240 - acc: 0.9453 - val_loss: 0.26
97 - val_acc: 0.9260
Epoch 29/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1271 - acc: 0.9444 - val_loss: 0.31
04 - val_acc: 0.9284
Epoch 30/30
7352/7352 [==============================] - 10s 1ms/step - loss: 0.1243 - acc: 0.9476 - val_loss: 0.32
17 - val_acc: 0.9213
Test loss: 0.3216777176025079
Test accuracy: 0.9212758737699356
```

Test accuracy: 0.9212758737699356

In [116]:

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

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

out_table.add_row(["CNN - relu","128 kernal=7","80%",score])
print(out_table)
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 537        0         0        0                   0
SITTING                  0      374       109        0                   0
STANDING                 0       70       457        0                   0
WALKING                  0        0         0      495                   0
WALKING_DOWNSTAIRS       0        0         0        9                 390
WALKING_UPSTAIRS         0        0         0        6                   3

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            8
STANDING                           5
WALKING                            1
WALKING_DOWNSTAIRS                21
WALKING_UPSTAIRS                 462
2947/2947 [==============================] - 1s 245us/step
[0.3216777176025079, 0.9212758737699356]
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                  MODEL                  |      Hidden unit      |  Dropout  |                  Accu
rcy                   |
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|               LSTM One                  |          32           |    50%    |         [0.6969475907
087326, 0.6]          |
|               LSTM One                  |          64           |    50%    |        [1.08183042234
18236, 0.566]         |
|               LSTM One                  |          64           |    70%    |        [0.91488204717
63611, 0.804]         |
|               LSTM Two                  |          64           |    80%    |        [1.13715102674
43954, 0.578]         |
|          LSTM One - Highdata            |          64           |    80%    |          [nan, 0.1683
06752629793]          |
|       LSTM One - Highdata + Relu        |          64           |    70%    |          [nan, 0.1683
06752629793]          |
|     LSTM One - Highdata + sigmoid       |          64           |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |          64           |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|          LSTM Two - Highdata            |          32           |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|          LSTM Two - Highdata            |        128 - 64       | 50% - 70% |          [nan, 0.1683
06752629793]          |
|          LSTM Two - Highdata            |        128 - 64       | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                  CNN                    |        32 - 64        |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                  CNN                    |        64 - 128       |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                  CNN                    |   64 - 128 kernal=20  |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                  CNN                    | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                  CNN                    | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
|              CNN - sigmoid              |   64-128 kernal=5,7   |    70%    |  [1.793824812902367,
0.18221920597217509] |
|                CNN - relu               |   64-128 kernal=5,7   |    70%    |  [0.4498406055611254,
0.9216152019002375] |
|                CNN - relu               |   64-128 kernal=50,5  |    70%    |  [0.7203596894995468,
0.9178825924669155] |
```

```
|                CNN - relu                |   128-64 kernal=50,5   |    80%    |  [0.6793473407663981,
0.9226331862911435] |
|                CNN - relu                |    64-32 kernal=5,3    |    80%    |  [0.7431288106871484,
0.9056667797760435] |
|                CNN - relu                |      128 kernal=7      |    80%    |  [0.3216777176025079,
0.9212758737699356] |
+------------------------------------------+------------------------+-----------+----------------------
--------------------+
```

CNN Model10

In [117]:

```
model = Sequential()
model.add(Conv1D(256, kernel_size=7,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(Conv1D(128, 5, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.8))
model.add(Dense(n_classes, activation='softmax'))

model.summary()




model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_47 (Conv1D)           (None, 122, 256)          16384
_________________________________________________________________
conv1d_48 (Conv1D)           (None, 118, 128)          163968
_________________________________________________________________
max_pooling1d_20 (MaxPooling (None, 29, 128)           0
_________________________________________________________________
flatten_17 (Flatten)         (None, 3712)              0
_________________________________________________________________
dense_33 (Dense)             (None, 128)               475264
_________________________________________________________________
dropout_22 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_34 (Dense)             (None, 6)                 774
=================================================================
Total params: 656,390
Trainable params: 656,390
Non-trainable params: 0
_________________________________________________________________
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 51s 7ms/step - loss: 0.7564 - acc: 0.6842 - val_loss: 0.31
02 - val_acc: 0.8823
Epoch 2/30
7352/7352 [==============================] - 51s 7ms/step - loss: 0.3612 - acc: 0.8482 - val_loss: 0.37
79 - val_acc: 0.9074
Epoch 3/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.3007 - acc: 0.8776 - val_loss: 0.24
91 - val_acc: 0.9230
```

```
Epoch 4/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.2278 - acc: 0.8966 - val_loss: 0.49
01 - val_acc: 0.8999
Epoch 5/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.2578 - acc: 0.8999 - val_loss: 0.37
25 - val_acc: 0.9101
Epoch 6/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.1983 - acc: 0.9168 - val_loss: 0.39
59 - val_acc: 0.9050
Epoch 7/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.1948 - acc: 0.9204 - val_loss: 0.43
03 - val_acc: 0.9040
Epoch 8/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.1804 - acc: 0.9255 - val_loss: 0.41
44 - val_acc: 0.9162
Epoch 9/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1661 - acc: 0.9289 - val_loss: 0.98
01 - val_acc: 0.8772
Epoch 10/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.2150 - acc: 0.9187 - val_loss: 0.27
57 - val_acc: 0.9304
Epoch 11/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.2016 - acc: 0.9301 - val_loss: 0.26
97 - val_acc: 0.9264
Epoch 12/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1526 - acc: 0.9320 - val_loss: 0.24
34 - val_acc: 0.9318
Epoch 13/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.1592 - acc: 0.9302 - val_loss: 0.24
99 - val_acc: 0.9321
Epoch 14/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.1473 - acc: 0.9350 - val_loss: 0.52
85 - val_acc: 0.9128
Epoch 15/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.1743 - acc: 0.9344 - val_loss: 0.38
17 - val_acc: 0.9332
Epoch 16/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1929 - acc: 0.9274 - val_loss: 0.38
96 - val_acc: 0.9233
Epoch 17/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.2622 - acc: 0.9272 - val_loss: 0.43
61 - val_acc: 0.9287
Epoch 18/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.3324 - acc: 0.9264 - val_loss: 0.44
07 - val_acc: 0.9328
Epoch 19/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.2558 - acc: 0.9304 - val_loss: 0.40
85 - val_acc: 0.9352
Epoch 20/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.2387 - acc: 0.9320 - val_loss: 0.42
12 - val_acc: 0.9318
Epoch 21/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1774 - acc: 0.9408 - val_loss: 0.47
85 - val_acc: 0.9179
Epoch 22/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1636 - acc: 0.9391 - val_loss: 0.46
55 - val_acc: 0.9230
Epoch 23/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.1725 - acc: 0.9397 - val_loss: 0.47
05 - val_acc: 0.9277
Epoch 24/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1579 - acc: 0.9448 - val_loss: 0.55
69 - val_acc: 0.9155
Epoch 25/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.2693 - acc: 0.9344 - val_loss: 0.80
97 - val_acc: 0.9053
Epoch 26/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1906 - acc: 0.9433 - val_loss: 0.62
72 - val_acc: 0.9141
Epoch 27/30
7352/7352 [==============================] - 49s 7ms/step - loss: 0.1537 - acc: 0.9452 - val_loss: 0.47
73 - val_acc: 0.9308
Epoch 28/30
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1431 - acc: 0.9528 - val_loss: 0.71
29 - val_acc: 0.9223
Epoch 29/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1375 - acc: 0.9553 - val_loss: 0.70
```

```
07 - val_acc: 0.9233
Epoch 30/30
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1403 - acc: 0.9525 - val_loss: 0.64
16 - val_acc: 0.9155
Test loss: 0.6415796165012791
Test accuracy: 0.9155072955548015
```

In [118]:

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

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

out_table.add_row(["CNN - relu","256-128 kernal=7,5","80%",score])
print(out_table)
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 519        0         1        0                   0
SITTING                  0      375       107        0                   0
STANDING                 0       81       450        1                   0
WALKING                  0        0         0      490                   0
WALKING_DOWNSTAIRS       0        0         0        9                 403
WALKING_UPSTAIRS         0        4         0        1                   5

Pred                WALKING_UPSTAIRS
True
LAYING                            17
SITTING                            9
STANDING                           0
WALKING                            6
WALKING_DOWNSTAIRS                 8
WALKING_UPSTAIRS                 461
2947/2947 [==============================] - 5s 2ms/step
[0.6415796165012791, 0.9155072955548015]
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                  MODEL                  |      Hidden unit      |  Dropout  |                   Accu
rcy                   |
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                LSTM One                 |          32           |    50%    |          [0.6969475907
087326, 0.6]          |
|                LSTM One                 |          64           |    50%    |         [1.08183042234
18236, 0.566]         |
|                LSTM One                 |          64           |    70%    |         [0.91488204717
63611, 0.804]         |
|                LSTM Two                 |          64           |    80%    |         [1.13715102674
43954, 0.578]         |
|           LSTM One - Highdata           |          64           |    80%    |           [nan, 0.1683
06752629793]          |
|        LSTM One - Highdata + Relu       |          64           |    70%    |           [nan, 0.1683
06752629793]          |
|      LSTM One - Highdata + sigmoid      |          64           |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |          64           |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|           LSTM Two - Highdata           |          32           |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|           LSTM Two - Highdata           |        128 - 64       | 50% - 70% |           [nan, 0.1683
06752629793]          |
|           LSTM Two - Highdata           |        128 - 64       | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                CNN                      |        32 - 64        |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                CNN                      |        64 - 128       |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                CNN                      |   64 - 128 kernal=20  |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                CNN                      | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                CNN                      | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
```

```
|              CNN - sigmoid              |   64-128 kernal=5,7    |    70%    |  [1.7938248129023 67,
0.18221920597217509] |
|                CNN - relu               |   64-128 kernal=5,7    |    70%    |  [0.4498406055611254,
0.9216152019002375] |
|                CNN - relu               |   64-128 kernal=50,5   |    70%    |  [0.7203596894995468,
0.9178825924669155] |
|                CNN - relu               |   128-64 kernal=50,5   |    80%    |  [0.6793473407663981,
0.9226331862911435] |
|                CNN - relu               |    64-32 kernal=5,3    |    80%    |  [0.7431288106871484,
0.9056667797760435] |
|                CNN - relu               |      128 kernal=7      |    80%    |  [0.3216777176025079,
0.9212758737699356] |
|                CNN - relu               |   256-128 kernal=7,5   |    80%    |  [0.6415796165012791,
0.9155072955548015] |
+-----------------------------------------+------------------------+-----------+----------------------
---------------------+
```

CNN Model11

In [119]:

```
model = Sequential()
model.add(Conv1D(256, kernel_size=7,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.7))
model.add(Dense(n_classes, activation='softmax'))

model.summary()



model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_49 (Conv1D)           (None, 122, 256)          16384
_________________________________________________________________
max_pooling1d_21 (MaxPooling (None, 30, 256)           0
_________________________________________________________________
flatten_18 (Flatten)         (None, 7680)              0
_________________________________________________________________
dense_35 (Dense)             (None, 128)               983168
_________________________________________________________________
dropout_23 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_36 (Dense)             (None, 6)                 774
=================================================================
Total params: 1,000,326
Trainable params: 1,000,326
Non-trainable params: 0
_________________________________________________________________
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 20s 3ms/step - loss: 0.5606 - acc: 0.7790 - val_loss: 0.28
49 - val_acc: 0.8911
Epoch 2/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.2314 - acc: 0.9153 - val_loss: 0.20
```

```
32 - val_acc: 0.9284
Epoch 3/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1820 - acc: 0.9297 - val_loss: 0.19
02 - val_acc: 0.9270
Epoch 4/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1815 - acc: 0.9324 - val_loss: 0.31
09 - val_acc: 0.9230
Epoch 5/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1619 - acc: 0.9376 - val_loss: 0.29
45 - val_acc: 0.9141
Epoch 6/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1534 - acc: 0.9372 - val_loss: 0.47
66 - val_acc: 0.9019
Epoch 7/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1416 - acc: 0.9437 - val_loss: 0.35
21 - val_acc: 0.9155
Epoch 8/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1439 - acc: 0.9378 - val_loss: 0.33
28 - val_acc: 0.9172
Epoch 9/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1483 - acc: 0.9406 - val_loss: 0.25
57 - val_acc: 0.9172
Epoch 10/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1295 - acc: 0.9478 - val_loss: 0.22
89 - val_acc: 0.9348
Epoch 11/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1274 - acc: 0.9463 - val_loss: 0.31
10 - val_acc: 0.9152
Epoch 12/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1257 - acc: 0.9446 - val_loss: 0.23
60 - val_acc: 0.9230
Epoch 13/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1336 - acc: 0.9463 - val_loss: 0.31
04 - val_acc: 0.9165
Epoch 14/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1243 - acc: 0.9442 - val_loss: 0.27
46 - val_acc: 0.9308
Epoch 15/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1247 - acc: 0.9465 - val_loss: 0.30
09 - val_acc: 0.9206
Epoch 16/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1291 - acc: 0.9460 - val_loss: 0.22
83 - val_acc: 0.9253
Epoch 17/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1173 - acc: 0.9476 - val_loss: 0.33
43 - val_acc: 0.9087
Epoch 18/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1174 - acc: 0.9517 - val_loss: 0.28
33 - val_acc: 0.9196
Epoch 19/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.1089 - acc: 0.9476 - val_loss: 0.30
56 - val_acc: 0.9311
Epoch 20/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1078 - acc: 0.9489 - val_loss: 0.31
32 - val_acc: 0.9270
Epoch 21/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1178 - acc: 0.9457 - val_loss: 0.27
44 - val_acc: 0.9199
Epoch 22/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1054 - acc: 0.9514 - val_loss: 0.25
99 - val_acc: 0.9281
Epoch 23/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1063 - acc: 0.9506 - val_loss: 0.31
20 - val_acc: 0.9277068
Epoch 24/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1114 - acc: 0.9497 - val_loss: 0.33
67 - val_acc: 0.9220
Epoch 25/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1030 - acc: 0.9520 - val_loss: 0.33
42 - val_acc: 0.9376
Epoch 26/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.0986 - acc: 0.9550 - val_loss: 0.29
55 - val_acc: 0.9284
Epoch 27/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1055 - acc: 0.9528 - val_loss: 0.30
83 - val_acc: 0.9274
Epoch 28/30
```

```
7352/7352 [==============================] - 17s 2ms/step - loss: 0.0911 - acc: 0.9548 - val_loss: 0.35
51 - val_acc: 0.9369
Epoch 29/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.0960 - acc: 0.9557 - val_loss: 0.44
43 - val_acc: 0.9199
Epoch 30/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.0861 - acc: 0.9581 - val_loss: 0.34
43 - val_acc: 0.9359
Test loss: 0.3442622534280121
Test accuracy: 0.9358669833729216
```

In [120]:

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

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

out_table.add_row(["CNN - relu","256 kernal=7","70%",score])
print(out_table)
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 537        0         0        0                   0
SITTING                  0      408        69        0                   0
STANDING                 0       89       441        0                   0
WALKING                  0        0         0      494                   1
WALKING_DOWNSTAIRS       0        0         0        1                 417
WALKING_UPSTAIRS         0        0         0        0                  10

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                           14
STANDING                           2
WALKING                            1
WALKING_DOWNSTAIRS                 2
WALKING_UPSTAIRS                 461
2947/2947 [==============================] - 1s 400us/step
[0.3442622534280121, 0.9358669833729216]
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                  MODEL                  |      Hidden unit      |  Dropout  |                   Accu
rcy                   |
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                LSTM One                 |          32           |    50%    |         [0.6969475907
087326, 0.6]          |
|                LSTM One                 |          64           |    50%    |        [1.08183042234
18236, 0.566]         |
|                LSTM One                 |          64           |    70%    |        [0.91488204717
63611, 0.804]         |
|                LSTM Two                 |          64           |    80%    |        [1.13715102674
43954, 0.578]         |
|           LSTM One - Highdata           |          64           |    80%    |          [nan, 0.1683
06752629793]          |
|        LSTM One - Highdata + Relu       |          64           |    70%    |          [nan, 0.1683
06752629793]          |
|      LSTM One - Highdata + sigmoid      |          64           |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |          64           |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|           LSTM Two - Highdata           |          32           |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|           LSTM Two - Highdata           |        128 - 64       | 50% - 70% |          [nan, 0.1683
06752629793]          |
|           LSTM Two - Highdata           |        128 - 64       | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                CNN                      |        32 - 64        |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                CNN                      |        64 - 128       |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                CNN                      |   64 - 128 kernal=20  |    50%    | [0.48552406951467336,
0.9175432643366135] |
```

```
|                   CNN                    | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                   CNN                    | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
|              CNN - sigmoid               |   64-128 kernal=5,7    |    70%    |  [1.793824812902367,
0.18221920597217509] |
|                CNN - relu                |   64-128 kernal=5,7    |    70%    |  [0.4498406055611254,
0.9216152019002375] |
|                CNN - relu                |   64-128 kernal=50,5   |    70%    |  [0.7203596894995468,
0.9178825924669155] |
|                CNN - relu                |   128-64 kernal=50,5   |    80%    |  [0.6793473407663981,
0.9226331862911435] |
|                CNN - relu                |    64-32 kernal=5,3    |    80%    |  [0.7431288106871484,
0.9056667797760435] |
|                CNN - relu                |      128 kernal=7      |    80%    |  [0.3216777176025079,
0.9212758737699356] |
|                CNN - relu                |   256-128 kernal=7,5   |    80%    |  [0.6415796165012791,
0.9155072955548015] |
|                CNN - relu                |      256 kernal=7      |    70%    |  [0.3442622534280121,
0.9358669833729216] |
+------------------------------------------+------------------------+-----------+----------------------
--------------------+
```

CNN Model12

In [121]:

```
model = Sequential()
model.add(Conv1D(512, kernel_size=7,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.7))
model.add(Dense(n_classes, activation='softmax'))

model.summary()



model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_50 (Conv1D)           (None, 122, 512)          32768
_________________________________________________________________
max_pooling1d_22 (MaxPooling (None, 30, 512)           0
_________________________________________________________________
flatten_19 (Flatten)         (None, 15360)             0
_________________________________________________________________
dense_37 (Dense)             (None, 128)               1966208
_________________________________________________________________
dropout_24 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_38 (Dense)             (None, 6)                 774
=================================================================
Total params: 1,999,750
Trainable params: 1,999,750
Non-trainable params: 0
```

```
_________________________________________________________________
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 38s 5ms/step - loss: 0.5482 - acc: 0.7856 - val_loss: 0.30
24 - val_acc: 0.8931
Epoch 2/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.2575 - acc: 0.9033 - val_loss: 0.21
56 - val_acc: 0.9121
Epoch 3/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1956 - acc: 0.9215 - val_loss: 0.38
70 - val_acc: 0.9053
Epoch 4/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1794 - acc: 0.9272 - val_loss: 0.41
35 - val_acc: 0.9135
Epoch 5/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1861 - acc: 0.9302 - val_loss: 0.28
68 - val_acc: 0.9220
Epoch 6/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1545 - acc: 0.9382 - val_loss: 0.24
86 - val_acc: 0.9213
Epoch 7/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1560 - acc: 0.9363 - val_loss: 0.34
71 - val_acc: 0.9206
Epoch 8/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1590 - acc: 0.9377 - val_loss: 0.36
75 - val_acc: 0.9104
Epoch 9/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1785 - acc: 0.9346 - val_loss: 0.39
31 - val_acc: 0.9152
Epoch 10/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1332 - acc: 0.9436 - val_loss: 0.42
38 - val_acc: 0.9216
Epoch 11/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1277 - acc: 0.9450 - val_loss: 0.48
25 - val_acc: 0.9162
Epoch 12/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1410 - acc: 0.9404 - val_loss: 0.47
46 - val_acc: 0.9091
Epoch 13/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1304 - acc: 0.9440 - val_loss: 0.46
49 - val_acc: 0.9121
Epoch 14/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1388 - acc: 0.9419 - val_loss: 0.47
19 - val_acc: 0.9070
Epoch 15/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1259 - acc: 0.9479 - val_loss: 0.53
66 - val_acc: 0.9067
Epoch 16/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1259 - acc: 0.9455 - val_loss: 0.59
28 - val_acc: 0.8999
Epoch 17/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1183 - acc: 0.9478 - val_loss: 0.51
40 - val_acc: 0.9030
Epoch 18/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1377 - acc: 0.9452 - val_loss: 0.56
18 - val_acc: 0.9009
Epoch 19/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1376 - acc: 0.9448 - val_loss: 0.61
59 - val_acc: 0.8972
Epoch 20/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1135 - acc: 0.9474 - val_loss: 0.41
67 - val_acc: 0.9152
Epoch 21/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1113 - acc: 0.9520 - val_loss: 0.35
02 - val_acc: 0.9158
Epoch 22/30
7352/7352 [==============================] - 37s 5ms/step - loss: 0.1087 - acc: 0.9506 - val_loss: 0.54
00 - val_acc: 0.9101
Epoch 23/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1058 - acc: 0.9495 - val_loss: 0.45
23 - val_acc: 0.9141
Epoch 24/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.0991 - acc: 0.9501 - val_loss: 0.42
91 - val_acc: 0.9155
Epoch 25/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.1114 - acc: 0.9484 - val_loss: 0.62
58 - val_acc: 0.9013
Epoch 26/30
```

```
Epoch 26/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1403 - acc: 0.9452 - val_loss: 0.53
36 - val_acc: 0.9189
Epoch 27/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1295 - acc: 0.9495 - val_loss: 0.43
89 - val_acc: 0.9233
Epoch 28/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1335 - acc: 0.9470 - val_loss: 0.50
61 - val_acc: 0.9216
Epoch 29/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.1268 - acc: 0.9499 - val_loss: 0.43
39 - val_acc: 0.9152
Epoch 30/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.0935 - acc: 0.9504 - val_loss: 0.45
32 - val_acc: 0.9240
Test loss: 0.4532000694376894
Test accuracy: 0.9239904988123515
```

In [122]:

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

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

out_table.add_row(["CNN - relu","512 kernal=7","70%",score])
print(out_table)
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 524        0         0        0                   0
SITTING                  0      414        73        0                   0
STANDING                 0      109       422        0                   0
WALKING                  0        0         0      488                   2
WALKING_DOWNSTAIRS       0        1         0        0                 416
WALKING_UPSTAIRS         0        0         0        0                  12

Pred                WALKING_UPSTAIRS
True
LAYING                            13
SITTING                            4
STANDING                           1
WALKING                            6
WALKING_DOWNSTAIRS                 3
WALKING_UPSTAIRS                 459
2947/2947 [==============================] - 2s 750us/step
[0.4532000694376894, 0.9239904988123515]
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                  MODEL                  |      Hidden unit      |  Dropout  |                  Accu
rcy                 |
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                LSTM One                 |          32           |    50%    |         [0.6969475907
087326, 0.6]        |
|                LSTM One                 |          64           |    50%    |        [1.08183042234
18236, 0.566]       |
|                LSTM One                 |          64           |    70%    |        [0.91488204717
63611, 0.804]       |
|                LSTM Two                 |          64           |    80%    |        [1.13715102674
43954, 0.578]       |
|           LSTM One - Highdata           |          64           |    80%    |          [nan, 0.1683
06752629793]        |
|        LSTM One - Highdata + Relu       |          64           |    70%    |          [nan, 0.1683
06752629793]        |
|      LSTM One - Highdata + sigmoid      |          64           |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |          64           |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|           LSTM Two - Highdata           |          32           |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|           LSTM Two - Highdata           |        128 - 64       | 50% - 70% |          [nan, 0.1683
06752629793]        |
|           LSTM Two - Highdata           |        128 - 64       | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
```

```
|                  CNN                    |        32 - 64         |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                  CNN                    |        64 - 128        |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                  CNN                    |   64 - 128 kernal=20   |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                  CNN                    | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                  CNN                    | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
|              CNN - sigmoid              |   64-128 kernal=5,7    |    70%    |  [1.793824812902367,
0.18221920597217509] |
|                CNN - relu               |   64-128 kernal=5,7    |    70%    |  [0.4498406055611254,
0.9216152019002375] |
|                CNN - relu               |   64-128 kernal=50,5   |    70%    |  [0.7203596894995468,
0.9178825924669155] |
|                CNN - relu               |   128-64 kernal=50,5   |    80%    |  [0.6793473407663981,
0.9226331862911435] |
|                CNN - relu               |    64-32 kernal=5,3    |    80%    |  [0.7431288106871484,
0.9056667797760435] |
|                CNN - relu               |      128 kernal=7      |    80%    |  [0.3216777176025079,
0.9212758737699356] |
|                CNN - relu               |   256-128 kernal=7,5   |    80%    |  [0.6415796165012791,
0.9155072955548015] |
|                CNN - relu               |      256 kernal=7      |    70%    |  [0.3442622534280121,
0.9358669833729216] |
|                CNN - relu               |      512 kernal=7      |    70%    |  [0.4532000694376894,
0.9239904988123515] |
+-----------------------------------------+------------------------+-----------+---------------------
--------------------+
```

CNN Model 12

We will try decreasing the Kernal and observe the performance

In [123]:

```
model = Sequential()
model.add(Conv1D(256, kernel_size=5,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.7))
model.add(Dense(n_classes, activation='softmax'))

model.summary()




model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_51 (Conv1D)           (None, 124, 256)          11776
_________________________________________________________________
max_pooling1d_23 (MaxPooling (None, 31, 256)           0
_________________________________________________________________
flatten_20 (Flatten)         (None, 7936)              0
```

```
flatten_20 (Flatten)         (None, 7936)              0
_________________________________________________________________
dense_39 (Dense)             (None, 128)               1015936
_________________________________________________________________
dropout_25 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_40 (Dense)             (None, 6)                 774
=================================================================
Total params: 1,028,486
Trainable params: 1,028,486
Non-trainable params: 0
_________________________________________________________________
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 20s 3ms/step - loss: 0.5837 - acc: 0.7671 - val_loss: 0.44
60 - val_acc: 0.8622
Epoch 2/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.2614 - acc: 0.9036 - val_loss: 0.29
00 - val_acc: 0.8965
Epoch 3/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.2113 - acc: 0.9202 - val_loss: 0.33
66 - val_acc: 0.8975
Epoch 4/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1824 - acc: 0.9263 - val_loss: 0.24
69 - val_acc: 0.9077
Epoch 5/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1638 - acc: 0.9328 - val_loss: 0.22
65 - val_acc: 0.9101
Epoch 6/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1527 - acc: 0.9392 - val_loss: 0.27
88 - val_acc: 0.9138
Epoch 7/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1497 - acc: 0.9389 - val_loss: 0.32
65 - val_acc: 0.9036
Epoch 8/30
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1552 - acc: 0.9365 - val_loss: 0.44
58 - val_acc: 0.8945
Epoch 9/30
7352/7352 [==============================] - 20s 3ms/step - loss: 0.1415 - acc: 0.9415 - val_loss: 0.39
25 - val_acc: 0.9067
Epoch 10/30
7352/7352 [==============================] - 23s 3ms/step - loss: 0.1281 - acc: 0.9456 - val_loss: 0.39
24 - val_acc: 0.9111
Epoch 11/30
7352/7352 [==============================] - 23s 3ms/step - loss: 0.1325 - acc: 0.9450 - val_loss: 0.40
71 - val_acc: 0.9121
Epoch 12/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.1372 - acc: 0.9438 - val_loss: 0.43
26 - val_acc: 0.9033
Epoch 13/30
7352/7352 [==============================] - 23s 3ms/step - loss: 0.1283 - acc: 0.9452 - val_loss: 0.45
19 - val_acc: 0.9131
Epoch 14/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.1283 - acc: 0.9422 - val_loss: 0.23
75 - val_acc: 0.9226
Epoch 15/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.1230 - acc: 0.9450 - val_loss: 0.24
83 - val_acc: 0.9138
Epoch 16/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.1189 - acc: 0.9470 - val_loss: 0.32
27 - val_acc: 0.9213
Epoch 17/30
7352/7352 [==============================] - 23s 3ms/step - loss: 0.1126 - acc: 0.9498 - val_loss: 0.28
37 - val_acc: 0.9169
Epoch 18/30
7352/7352 [==============================] - 22s 3ms/step - loss: 0.1156 - acc: 0.9456 - val_loss: 0.48
37 - val_acc: 0.9067
Epoch 19/30
7352/7352 [==============================] - 21s 3ms/step - loss: 0.1112 - acc: 0.9484 - val_loss: 0.34
70 - val_acc: 0.9067
Epoch 20/30
7352/7352 [==============================] - 22s 3ms/step - loss: 0.1134 - acc: 0.9514 - val_loss: 0.32
68 - val_acc: 0.9104
Epoch 21/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1040 - acc: 0.9535 - val_loss: 0.33
85 - val_acc: 0.9243
Epoch 22/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1134 - acc: 0.9480 - val_loss: 0.32
```

```
35 - val_acc: 0.9186
Epoch 23/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1313 - acc: 0.9502 - val_loss: 0.37
06 - val_acc: 0.9203
Epoch 24/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1021 - acc: 0.9524 - val_loss: 0.32
28 - val_acc: 0.9270
Epoch 25/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1017 - acc: 0.9543 - val_loss: 0.38
76 - val_acc: 0.9138
Epoch 26/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1037 - acc: 0.9532 - val_loss: 0.32
23 - val_acc: 0.9223
Epoch 27/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1109 - acc: 0.9509 - val_loss: 0.27
59 - val_acc: 0.9216
Epoch 28/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1010 - acc: 0.9528 - val_loss: 0.31
80 - val_acc: 0.9274
Epoch 29/30
7352/7352 [==============================] - 17s 2ms/step - loss: 0.0998 - acc: 0.9499 - val_loss: 0.33
67 - val_acc: 0.9267
Epoch 30/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.0929 - acc: 0.9566 - val_loss: 0.43
68 - val_acc: 0.9192
Test loss: 0.4368252168530802
Test accuracy: 0.9192399049881235
```

In [124]:

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

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

out_table.add_row(["CNN - relu","256 kernal=5","70%",score])
print(out_table)
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 537        0         0        0                   0
SITTING                  0      381        85        0                   0
STANDING                 0       54       476        0                   0
WALKING                  0        0         0      494                   1
WALKING_DOWNSTAIRS       0        0         0       23                 397
WALKING_UPSTAIRS         0        0         0       33                  14

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                           25
STANDING                           2
WALKING                            1
WALKING_DOWNSTAIRS                 0
WALKING_UPSTAIRS                 424
2947/2947 [==============================] - 1s 404us/step
[0.4368252168530802, 0.9192399049881235]
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                  MODEL                  |      Hidden unit      |  Dropout  |                  Accu
rcy                 |
+-----------------------------------------+-----------------------+-----------+----------------------
---------------------+
|                LSTM One                 |          32           |    50%    |         [0.6969475907
087326, 0.6]        |
|                LSTM One                 |          64           |    50%    |        [1.08183042234
18236, 0.566]       |
|                LSTM One                 |          64           |    70%    |        [0.91488204717
63611, 0.804]       |
|                LSTM Two                 |          64           |    80%    |        [1.13715102674
43954, 0.578]       |
|           LSTM One - Highdata           |          64           |    80%    |          [nan, 0.1683
06752629793]        |
|        LSTM One - Highdata + Relu       |          64           |    70%    |          [nan, 0.1683
```

```
06752629793]         |
|      LSTM One - Highdata + sigmoid       |           64           |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |           64           |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|           LSTM Two - Highdata            |           32           |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|           LSTM Two - Highdata            |        128 - 64        | 50% - 70% |          [nan, 0.1683
06752629793]         |
|           LSTM Two - Highdata            |        128 - 64        | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                  CNN                     |        32 - 64         |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                  CNN                     |        64 - 128        |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                  CNN                     |   64 - 128 kernal=20   |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                  CNN                     | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                  CNN                     | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
|              CNN - sigmoid               |   64-128 kernal=5,7    |    70%    |  [1.793824812902367,
0.18221920597217509] |
|                CNN - relu                |   64-128 kernal=5,7    |    70%    |  [0.4498406055611254,
0.9216152019002375] |
|                CNN - relu                |   64-128 kernal=50,5   |    70%    |  [0.7203596894995468,
0.9178825924669155] |
|                CNN - relu                |   128-64 kernal=50,5   |    80%    |  [0.6793473407663981,
0.9226331862911435] |
|                CNN - relu                |    64-32 kernal=5,3    |    80%    |  [0.7431288106871484,
0.9056667797760435] |
|                CNN - relu                |      128 kernal=7      |    80%    |  [0.3216777176025079,
0.9212758737699356] |
|                CNN - relu                |   256-128 kernal=7,5   |    80%    |  [0.6415796165012791,
0.9155072955548015] |
|                CNN - relu                |      256 kernal=7      |    70%    |  [0.3442622534280121,
0.9358669833729216] |
|                CNN - relu                |      512 kernal=7      |    70%    |  [0.4532000694376894,
0.9239904988123515] |
|                CNN - relu                |      256 kernal=5      |    70%    |  [0.4368252168530802,
0.9192399049881235] |
+------------------------------------------+------------------------+-----------+----------------------
---------------------+
```

CNN Model

In [125]:

```
model = Sequential()
model.add(Conv1D(256, kernel_size=9,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.7))
model.add(Dense(n_classes, activation='softmax'))

model.summary()




model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=30,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
```

```
print('Test accuracy:', score[1])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_52 (Conv1D)           (None, 120, 256)          20992
_________________________________________________________________
max_pooling1d_24 (MaxPooling (None, 30, 256)           0
_________________________________________________________________
flatten_21 (Flatten)         (None, 7680)              0
_________________________________________________________________
dense_41 (Dense)             (None, 128)               983168
_________________________________________________________________
dropout_26 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_42 (Dense)             (None, 6)                 774
=================================================================
Total params: 1,004,934
Trainable params: 1,004,934
Non-trainable params: 0
_________________________________________________________________
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 21s 3ms/step - loss: 0.5762 - acc: 0.7677 - val_loss: 0.26
31 - val_acc: 0.8968
Epoch 2/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.2354 - acc: 0.9079 - val_loss: 0.25
93 - val_acc: 0.8985
Epoch 3/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1894 - acc: 0.9215 - val_loss: 0.30
13 - val_acc: 0.9125
Epoch 4/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1807 - acc: 0.9275 - val_loss: 0.31
26 - val_acc: 0.9240
Epoch 5/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1805 - acc: 0.9291 - val_loss: 0.36
13 - val_acc: 0.9152
Epoch 6/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1526 - acc: 0.9363 - val_loss: 0.20
27 - val_acc: 0.9270
Epoch 7/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1534 - acc: 0.9378 - val_loss: 0.20
81 - val_acc: 0.9203
Epoch 8/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1470 - acc: 0.9359 - val_loss: 0.21
86 - val_acc: 0.9301
Epoch 9/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1433 - acc: 0.9416 - val_loss: 0.26
14 - val_acc: 0.9230
Epoch 10/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1580 - acc: 0.9404 - val_loss: 0.21
51 - val_acc: 0.9162
Epoch 11/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1260 - acc: 0.9444 - val_loss: 0.20
86 - val_acc: 0.9362
Epoch 12/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1297 - acc: 0.9423 - val_loss: 0.22
98 - val_acc: 0.9257
Epoch 13/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1244 - acc: 0.9446 - val_loss: 0.24
80 - val_acc: 0.9199
Epoch 14/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1313 - acc: 0.9434 - val_loss: 0.28
62 - val_acc: 0.9226
Epoch 15/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1126 - acc: 0.9478 - val_loss: 0.29
37 - val_acc: 0.9250
Epoch 16/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1123 - acc: 0.9483 - val_loss: 0.37
64 - val_acc: 0.9216
Epoch 17/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1054 - acc: 0.9499 - val_loss: 0.45
23 - val_acc: 0.9233
Epoch 18/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1079 - acc: 0.9498 - val_loss: 0.52
26 - val_acc: 0.9111
```

```
Epoch 19/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1080 - acc: 0.9513 - val_loss: 0.39
37 - val_acc: 0.9172
Epoch 20/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1268 - acc: 0.9470 - val_loss: 0.37
95 - val_acc: 0.9315
Epoch 21/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1013 - acc: 0.9506 - val_loss: 0.41
09 - val_acc: 0.9162
Epoch 22/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1025 - acc: 0.9525 - val_loss: 0.51
48 - val_acc: 0.9165
Epoch 23/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1044 - acc: 0.9516 - val_loss: 0.50
25 - val_acc: 0.9169
Epoch 24/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.0915 - acc: 0.9532 - val_loss: 0.57
09 - val_acc: 0.8979
Epoch 25/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1047 - acc: 0.9521 - val_loss: 0.59
03 - val_acc: 0.9080
Epoch 26/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1062 - acc: 0.9527 - val_loss: 0.55
47 - val_acc: 0.9223
Epoch 27/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.0951 - acc: 0.9569 - val_loss: 0.65
99 - val_acc: 0.9108
Epoch 28/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.0963 - acc: 0.9510 - val_loss: 0.53
73 - val_acc: 0.9233
Epoch 29/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.0901 - acc: 0.9550 - val_loss: 0.56
89 - val_acc: 0.9172
Epoch 30/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1028 - acc: 0.9550 - val_loss: 0.70
90 - val_acc: 0.9091
Test loss: 0.7089927847617913
Test accuracy: 0.9090600610790635
```

In [126]:

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

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

out_table.add_row(["CNN - relu","256 kernal=9","70%",score])
print(out_table)
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 510        0         0        0                   0
SITTING                  0      415        65        0                   0
STANDING                 0      104       427        1                   0
WALKING                  0        0         0      464                  24
WALKING_DOWNSTAIRS       0        0         0        0                 420
WALKING_UPSTAIRS         0        0         0        0                  28

Pred                WALKING_UPSTAIRS
True
LAYING                            27
SITTING                           11
STANDING                           0
WALKING                            8
WALKING_DOWNSTAIRS                 0
WALKING_UPSTAIRS                 443
2947/2947 [==============================] - 1s 483us/step
[0.7089927847617913, 0.9090600610790635]
+-----------------------------------------+----------------------+-----------+----------------------
--------------------+
|                MODEL                    |      Hidden unit     |  Dropout  |                  Accu
rcy                 |
+-----------------------------------------+----------------------+-----------+----------------------
--------------------+
|              LSTM One                   |          32          |    50%    |         [0.6969475907
```

```
|                LSTM One                |            32            |    50%    |          [0.6969473907
087326, 0.6]          |
|                LSTM One                |            64            |    50%    |         [1.08183042234
18236, 0.566]         |
|                LSTM One                |            64            |    70%    |         [0.91488204717
63611, 0.804]         |
|                LSTM Two                |            64            |    80%    |         [1.13715102674
43954, 0.578]         |
|           LSTM One - Highdata          |            64            |    80%    |           [nan, 0.1683
06752629793]          |
|        LSTM One - Highdata + Relu      |            64            |    70%    |           [nan, 0.1683
06752629793]          |
|      LSTM One - Highdata + sigmoid     |            64            |    70%    | [0.43612632637826976,
0.9022734984730234] |
| LSTM One - Highdata + sigmoid + 30 epoch |            64            |    70%    |  [0.3628289201868073,
0.8998982015609094] |
|           LSTM Two - Highdata          |            32            |    70%    |  [0.5005468663244095,
0.8856464200882254] |
|           LSTM Two - Highdata          |        128 - 64          | 50% - 70% |           [nan, 0.1683
06752629793]          |
|           LSTM Two - Highdata          |        128 - 64          | 30% - 50% |  [0.3769736843092622,
0.9178825924669155] |
|                  CNN                   |        32 - 64           |    50%    |  [0.6163437596247316,
0.9077027485578555] |
|                  CNN                   |        64 - 128          |    50%    |  [0.4507699583859416,
0.9168646080760094] |
|                  CNN                   |   64 - 128 kernal=20     |    50%    | [0.48552406951467336,
0.9175432643366135] |
|                  CNN                   | 32-64-128 kernal=3,5,5 |    50%    |  [0.529261440275583,
0.9127926705123854]  |
|                  CNN                   | 64-64-128 kernal=5,7,7 |    50%    |  [0.7584496770992061,
0.8998982015609094] |
|              CNN - sigmoid             |   64-128 kernal=5,7    |    70%    |  [1.793824812902367,
0.18221920597217509] |
|                CNN - relu              |   64-128 kernal=5,7    |    70%    |  [0.4498406055611254,
0.9216152019002375] |
|                CNN - relu              |   64-128 kernal=50,5   |    70%    |  [0.7203596894995468,
0.9178825924669155] |
|                CNN - relu              |   128-64 kernal=50,5   |    80%    |  [0.6793473407663981,
0.9226331862911435] |
|                CNN - relu              |    64-32 kernal=5,3    |    80%    |  [0.7431288106871484,
0.9056667797760435] |
|                CNN - relu              |      128 kernal=7      |    80%    |  [0.3216777176025079,
0.9212758737699356] |
|                CNN - relu              |   256-128 kernal=7,5   |    80%    |  [0.6415796165012791,
0.9155072955548015] |
|                CNN - relu              |      256 kernal=7      |    70%    |  [0.3442622534280121,
0.9358669833729216] |
|                CNN - relu              |      512 kernal=7      |    70%    |  [0.4532000694376894,
0.9239904988123515] |
|                CNN - relu              |      256 kernal=5      |    70%    |  [0.4368252168530802,
0.9192399049881235] |
|                CNN - relu              |      256 kernal=9      |    70%    |  [0.7089927847617913,
0.9090600610790635] |
+----------------------------------------+------------------------+-----------+----------------------
---------------------+
```

In [127]:

```
model = Sequential()
model.add(Conv1D(256, kernel_size=7,
                 activation='relu',
                 input_shape=(timesteps, input_dim)))
model.add(MaxPooling1D(pool_size=4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.7))
model.add(Dense(n_classes, activation='softmax'))

model.summary()




model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])
```

```
              metrics=['accuracy'])

history = model.fit(X_train, Y_train,
          batch_size=batch_size,
          epochs=40,
          verbose=1,
          validation_data=(X_test, Y_test))
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_53 (Conv1D)           (None, 122, 256)          16384
_________________________________________________________________
max_pooling1d_25 (MaxPooling (None, 30, 256)           0
_________________________________________________________________
flatten_22 (Flatten)         (None, 7680)              0
_________________________________________________________________
dense_43 (Dense)             (None, 128)               983168
_________________________________________________________________
dropout_27 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_44 (Dense)             (None, 6)                 774
=================================================================
Total params: 1,000,326
Trainable params: 1,000,326
Non-trainable params: 0
_________________________________________________________________
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 20s 3ms/step - loss: 0.5250 - acc: 0.8024 - val_loss: 0.25
77 - val_acc: 0.9002
Epoch 2/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.2201 - acc: 0.9199 - val_loss: 0.32
19 - val_acc: 0.9091
Epoch 3/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1853 - acc: 0.9308 - val_loss: 0.31
49 - val_acc: 0.9043
Epoch 4/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1582 - acc: 0.9393 - val_loss: 0.21
57 - val_acc: 0.9162
Epoch 5/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1538 - acc: 0.9389 - val_loss: 0.25
16 - val_acc: 0.9206
Epoch 6/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1548 - acc: 0.9400 - val_loss: 0.26
86 - val_acc: 0.9135
Epoch 7/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1499 - acc: 0.9441 - val_loss: 0.34
62 - val_acc: 0.9138
Epoch 8/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1513 - acc: 0.9410 - val_loss: 0.29
10 - val_acc: 0.9152
Epoch 9/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1377 - acc: 0.9472 - val_loss: 0.46
65 - val_acc: 0.9118
Epoch 10/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1360 - acc: 0.9452 - val_loss: 0.19
08 - val_acc: 0.9284
Epoch 11/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1256 - acc: 0.9489 - val_loss: 0.21
06 - val_acc: 0.9321
Epoch 12/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1210 - acc: 0.9489 - val_loss: 0.26
49 - val_acc: 0.9172
Epoch 13/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1340 - acc: 0.9431 - val_loss: 0.24
85 - val_acc: 0.9192
Epoch 14/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1301 - acc: 0.9460 - val_loss: 0.23
73 - val_acc: 0.9192
Epoch 15/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1205 - acc: 0.9490 - val_loss: 0.29
96 - val_acc: 0.9074
Epoch 16/30
```

```
Epoch 16/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1315 - acc: 0.9470 - val_loss: 0.22
58 - val_acc: 0.9335
Epoch 17/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1194 - acc: 0.9475 - val_loss: 0.32
15 - val_acc: 0.9121
Epoch 18/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1151 - acc: 0.9517 - val_loss: 0.31
81 - val_acc: 0.9141
Epoch 19/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1112 - acc: 0.9517 - val_loss: 0.36
18 - val_acc: 0.9179
Epoch 20/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1173 - acc: 0.9472 - val_loss: 0.38
46 - val_acc: 0.9077
Epoch 21/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1102 - acc: 0.9525 - val_loss: 0.36
45 - val_acc: 0.9203
Epoch 22/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1074 - acc: 0.9508 - val_loss: 0.35
83 - val_acc: 0.9097
Epoch 23/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1103 - acc: 0.9514 - val_loss: 0.39
62 - val_acc: 0.9199
Epoch 24/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1068 - acc: 0.9504 - val_loss: 0.52
42 - val_acc: 0.9125
Epoch 25/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1061 - acc: 0.9527 - val_loss: 0.41
56 - val_acc: 0.9223
Epoch 26/30
7352/7352 [==============================] - 18s 3ms/step - loss: 0.1082 - acc: 0.9506 - val_loss: 0.52
03 - val_acc: 0.9226
Epoch 27/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1344 - acc: 0.9497 - val_loss: 0.43
84 - val_acc: 0.9141
Epoch 28/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1034 - acc: 0.9565 - val_loss: 0.47
10 - val_acc: 0.9199
Epoch 29/30
7352/7352 [==============================] - 18s 2ms/step - loss: 0.1236 - acc: 0.9514 - val_loss: 0.52
91 - val_acc: 0.9111
Epoch 30/30
7352/7352 [==============================] - 19s 3ms/step - loss: 0.1030 - acc: 0.9520 - val_loss: 0.34
45 - val_acc: 0.9325
Test loss: 0.3445060393561512
Test accuracy: 0.9324737020699015
```

In [128]:

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

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

out_table.add_row(["CNN - relu","256 kernal=7","70%",score])
print(out_table)
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 537        0         0        0                   0
SITTING                  0      401        82        0                   0
STANDING                 0       85       444        0                   0
WALKING                  0        0         0      492                   2
WALKING_DOWNSTAIRS       0        0         0        0                 419
WALKING_UPSTAIRS         0        2         0        1                  13

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            8
STANDING                           3
WALKING                            2
WALKING_DOWNSTAIRS                 1
WALKING_UPSTAIRS                 455
```

```
2947/2947 [==============================] - 1s 417us/step
[0.3445060393561512, 0.9324737020699015]
```

MODEL	Hidden unit	Dropout	Accuracy
LSTM One	32	50%	[0.6969475907087326, 0.6]
LSTM One	64	50%	[1.0818304223418236, 0.566]
LSTM One	64	70%	[0.9148820471763611, 0.804]
LSTM Two	64	80%	[1.1371510267443954, 0.578]
LSTM One - Highdata	64	80%	[nan, 0.16830675262979 3]
LSTM One - Highdata + Relu	64	70%	[nan, 0.16830675262979 3]
LSTM One - Highdata + sigmoid	64	70%	[0.43612632637826976, 0.9022734984730234]
LSTM One - Highdata + sigmoid + 30 epoch	64	70%	[0.3628289201868073, 0.8998982015609094]
LSTM Two - Highdata	32	70%	[0.5005468663244095, 0.8856464200882254]
LSTM Two - Highdata	128 - 64	50% - 70%	[nan, 0.16830675262979 3]
LSTM Two - Highdata	128 - 64	30% - 50%	[0.3769736843092622, 0.9178825924669155]
CNN	32 - 64	50%	[0.6163437596247316, 0.9077027485578555]
CNN	64 - 128	50%	[0.4507699583859416, 0.9168646080760094]
CNN	64 - 128 kernal=20	50%	[0.48552406951467336, 0.9175432643366135]
CNN	32-64-128 kernal=3,5,5	50%	[0.529261440275583, 0.9127926705123854]
CNN	64-64-128 kernal=5,7,7	50%	[0.7584496770992061, 0.8998982015609094]
CNN - sigmoid	64-128 kernal=5,7	70%	[1.793824812902367, 0.18221920597217509]
CNN - relu	64-128 kernal=5,7	70%	[0.4498406055611254, 0.9216152019002375]
CNN - relu	64-128 kernal=50,5	70%	[0.7203596894995468, 0.9178825924669155]
CNN - relu	128-64 kernal=50,5	80%	[0.6793473407663981, 0.9226331862911435]
CNN - relu	64-32 kernal=5,3	80%	[0.7431288106871484, 0.9056667797760435]
CNN - relu	128 kernal=7	80%	[0.3216777176025079, 0.9212758737699356]
CNN - relu	256-128 kernal=7,5	80%	[0.6415796165012791, 0.9155072955548015]
CNN - relu	256 kernal=7	70%	[0.3442622534280121, 0.9358669833729216]
CNN - relu	512 kernal=7	70%	[0.4532000694376894, 0.9239904988123515]
CNN - relu	256 kernal=5	70%	[0.4368252168530802, 0.9192399049881235]
CNN - relu	256 kernal=9	70%	[0.7089927847617913, 0.9090600610790635]
CNN - relu	256 kernal=7	70%	[0.3445060393561512, 0.9324737020699015]

Conclusions

- We tried several scenarios but could not get a accuracy of 96%, however we got an accuracy of 93% when we use CNN 1D with 256 filters
- We can see in improvement in sitting and standing outputs in case of accuracy improvement

In [ ]: