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}
        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.ht
    """
    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 [54]:
```
# Importing libraries
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers.core import Dense, Dropout
from keras.layers.normalization import BatchNormalization
```

In [12]:
```
# Initializing parameters
epochs = 30
batch_size = 16
n_hidden = 32
```

In [13]:
```
# Utility function to count the number of classes
def _count_classes(y):
    return len(set([tuple(category) for category in y]))
```

In [14]:
```
# Loading the train and test data
X_train, X_test, Y_train, Y_test = load_data()
```

```
C:\Users\Admin\Anaconda3\envs\tensorflow_env\lib\site-packages\ipykernel_launch
er.py:12: FutureWarning: Method .as_matrix will be removed in a future version.
Use .values instead.
  if sys.path[0] == '':
```

In [15]:
```
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 [16]:
```
# 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\Admin\Anaconda3\envs\tensorflow_env\lib\site-p
ackages\tensorflow\python\framework\op_def_library.py:263: colocate_with (from
tensorflow.python.framework.ops) is deprecated and will be removed in a future
version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From C:\Users\Admin\Anaconda3\envs\tensorflow_env\lib\site-p
ackages\keras\backend\tensorflow_backend.py:3445: calling dropout (from tensorf
low.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a fu
ture version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - kee
p_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 [17]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

In [18]:
```
# Training the model
history_32_rms = model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=epochs)
```

```
WARNING:tensorflow:From C:\Users\Admin\Anaconda3\envs\tensorflow_env\lib\site-p
ackages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from tensorflow.pytho
n.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 60s 8ms/step - loss: 1.3233 - acc:
0.4332 - val_loss: 1.1553 - val_acc: 0.4649
Epoch 2/30
7352/7352 [==============================] - 55s 7ms/step - loss: 1.0250 - acc:
0.5536 - val_loss: 1.0156 - val_acc: 0.5806
Epoch 3/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.8296 - acc:
0.6432 - val_loss: 0.7981 - val_acc: 0.6043
Epoch 4/30
7352/7352 [==============================] - 56s 8ms/step - loss: 0.7209 - acc:
0.6542 - val_loss: 0.7564 - val_acc: 0.5955
Epoch 5/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.6490 - acc:
0.6702 - val_loss: 0.7268 - val_acc: 0.6237
Epoch 6/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.6264 - acc:
0.6780 - val_loss: 0.7316 - val_acc: 0.6206
Epoch 7/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.5941 - acc:
0.7065 - val_loss: 0.7197 - val_acc: 0.7017
Epoch 8/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.5716 - acc:
0.7291 - val_loss: 0.7148 - val_acc: 0.7279
Epoch 9/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.5232 - acc:
0.7788 - val_loss: 0.6354 - val_acc: 0.7448
Epoch 10/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.4901 - acc:
0.7930 - val_loss: 0.6373 - val_acc: 0.7513
Epoch 11/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.4514 - acc:
0.8041 - val_loss: 0.6609 - val_acc: 0.7506
Epoch 12/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.4343 - acc:
0.8202 - val_loss: 0.6811 - val_acc: 0.7669
Epoch 13/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.4740 - acc:
0.8380 - val_loss: 0.5476 - val_acc: 0.8409
Epoch 14/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.3630 - acc:
0.8841 - val_loss: 0.6571 - val_acc: 0.8290
Epoch 15/30
7352/7352 [==============================] - 51s 7ms/step - loss: 0.3168 - acc:
```

```
0.9081 - val_loss: 0.5576 - val_acc: 0.8633
Epoch 16/30
7352/7352 [==============================] - 53s 7ms/step - loss: 0.2820 - acc:
0.9169 - val_loss: 0.5482 - val_acc: 0.8633
Epoch 17/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.2524 - acc:
0.9219 - val_loss: 0.4965 - val_acc: 0.8768
Epoch 18/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.2359 - acc:
0.9346 - val_loss: 0.8069 - val_acc: 0.8107
Epoch 19/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.2083 - acc:
0.9382 - val_loss: 0.6350 - val_acc: 0.8439
Epoch 20/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1900 - acc:
0.9402 - val_loss: 0.7828 - val_acc: 0.8442
Epoch 21/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1966 - acc:
0.9391 - val_loss: 0.4236 - val_acc: 0.8850
Epoch 22/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1937 - acc:
0.9402 - val_loss: 0.4633 - val_acc: 0.8846
Epoch 23/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.2112 - acc:
0.9376 - val_loss: 0.5757 - val_acc: 0.8633
Epoch 24/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1882 - acc:
0.9389 - val_loss: 0.4698 - val_acc: 0.8748
Epoch 25/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1704 - acc:
0.9436 - val_loss: 0.6065 - val_acc: 0.8592
Epoch 26/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1592 - acc:
0.9453 - val_loss: 0.8993 - val_acc: 0.8558
Epoch 27/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1836 - acc:
0.9426 - val_loss: 0.5185 - val_acc: 0.9013
Epoch 28/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1672 - acc:
0.9431 - val_loss: 0.4778 - val_acc: 0.9013
Epoch 29/30
7352/7352 [==============================] - 53s 7ms/step - loss: 0.1754 - acc:
0.9480 - val_loss: 0.6840 - val_acc: 0.8741
Epoch 30/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1837 - acc:
0.9436 - val_loss: 0.5227 - val_acc: 0.8914
```

In [19]:
```
# Confusion Matrix
print(confusion_matrix(Y_test, model.predict(X_test)))
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 510        0         3        0                   0
SITTING                  0      416        72        0                   0
STANDING                 0      109       423        0                   0
WALKING                  0        0         0      448                  25
WALKING_DOWNSTAIRS       0        0         0        8                 392
WALKING_UPSTAIRS         0        1         1       15                  16

Pred                WALKING_UPSTAIRS
True
LAYING                            24
SITTING                            3
STANDING                           0
WALKING                           23
WALKING_DOWNSTAIRS                20
WALKING_UPSTAIRS                 438
```

In [20]:
```
score_32_rms = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 2s 565us/step
```

In [21]:
```
score_32_rms
```

Out[21]:
```
[0.5226698747499483, 0.8914149983033594]
```

LSTM with 48 units

In [22]:
```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(48, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model.add(Dropout(0.6))
# 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, 48)                11136
_________________________________________________________________
dropout_2 (Dropout)          (None, 48)                0
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 294
=================================================================
Total params: 11,430
Trainable params: 11,430
Non-trainable params: 0
_________________________________________________________________
```

In [23]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
optimizer='adam',
metrics=['accuracy'])
```

In [24]:
```
# Training the model
history_48_adam = 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 [==============================] - 57s 8ms/step - loss: 1.4184 - acc:
0.3617 - val_loss: 1.3882 - val_acc: 0.3773
Epoch 2/30
7352/7352 [==============================] - 55s 7ms/step - loss: 1.2866 - acc:
0.4248 - val_loss: 1.1935 - val_acc: 0.5185
Epoch 3/30
7352/7352 [==============================] - 55s 7ms/step - loss: 1.1797 - acc:
0.4710 - val_loss: 1.1463 - val_acc: 0.4649
Epoch 4/30
7352/7352 [==============================] - 55s 8ms/step - loss: 0.9862 - acc:
0.5147 - val_loss: 1.0529 - val_acc: 0.4863
Epoch 5/30
7352/7352 [==============================] - 60s 8ms/step - loss: 0.8807 - acc:
0.5569 - val_loss: 1.6132 - val_acc: 0.4242
Epoch 6/30
7352/7352 [==============================] - 56s 8ms/step - loss: 0.9183 - acc:
0.5520 - val_loss: 1.0211 - val_acc: 0.4866
Epoch 7/30
7352/7352 [==============================] - 55s 7ms/step - loss: 0.8614 - acc:
0.5928 - val_loss: 1.1428 - val_acc: 0.4876
Epoch 8/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.8884 - acc:
0.5646 - val_loss: 0.8936 - val_acc: 0.5687
Epoch 9/30
7352/7352 [==============================] - 55s 7ms/step - loss: 1.1494 - acc:
0.4695 - val_loss: 1.1175 - val_acc: 0.4795
Epoch 10/30
7352/7352 [==============================] - 55s 8ms/step - loss: 1.0399 - acc:
0.4974 - val_loss: 1.1226 - val_acc: 0.5704
Epoch 11/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.9371 - acc:
0.5953 - val_loss: 0.9779 - val_acc: 0.5738
Epoch 12/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.9313 - acc:
0.5540 - val_loss: 0.9783 - val_acc: 0.4920
Epoch 13/30
7352/7352 [==============================] - 55s 7ms/step - loss: 1.0694 - acc:
0.4897 - val_loss: 1.1942 - val_acc: 0.4506
Epoch 14/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.9081 - acc:
0.5763 - val_loss: 0.9156 - val_acc: 0.5891
Epoch 15/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.7414 - acc:
0.6459 - val_loss: 0.7864 - val_acc: 0.6098
Epoch 16/30
7352/7352 [==============================] - 55s 7ms/step - loss: 0.6810 - acc:
0.6600 - val_loss: 0.8112 - val_acc: 0.6115
Epoch 17/30
```

```
7352/7352 [==============================] - 56s 8ms/step - loss: 0.6747 - acc:
0.6563 - val_loss: 0.7723 - val_acc: 0.6149
Epoch 18/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.6579 - acc:
0.6628 - val_loss: 0.7784 - val_acc: 0.6135
Epoch 19/30
7352/7352 [==============================] - 55s 7ms/step - loss: 0.6465 - acc:
0.6654 - val_loss: 0.8052 - val_acc: 0.6091
Epoch 20/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.6508 - acc:
0.6575 - val_loss: 0.9754 - val_acc: 0.5826
Epoch 21/30
7352/7352 [==============================] - 55s 8ms/step - loss: 0.6854 - acc:
0.6575 - val_loss: 0.7546 - val_acc: 0.6105
Epoch 22/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.6522 - acc:
0.6620 - val_loss: 0.7014 - val_acc: 0.6145
Epoch 23/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.6165 - acc:
0.6676 - val_loss: 0.7881 - val_acc: 0.6183
Epoch 24/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.5987 - acc:
0.6760 - val_loss: 0.6246 - val_acc: 0.6203
Epoch 25/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.5540 - acc:
0.6844 - val_loss: 0.6213 - val_acc: 0.6244
Epoch 26/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.5609 - acc:
0.6912 - val_loss: 0.5818 - val_acc: 0.6373
Epoch 27/30
7352/7352 [==============================] - 55s 7ms/step - loss: 0.4910 - acc:
0.7242 - val_loss: 0.5404 - val_acc: 0.6827
Epoch 28/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.4795 - acc:
0.7675 - val_loss: 0.5253 - val_acc: 0.7526
Epoch 29/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.4208 - acc:
0.8177 - val_loss: 0.4409 - val_acc: 0.7737
Epoch 30/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.3608 - acc:
0.8334 - val_loss: 0.4317 - val_acc: 0.7648
```

In [25]:
```
# 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      395        85        8                   0
STANDING                 0      100       426        5                   0
WALKING                  0        0         0      460                   8
WALKING_DOWNSTAIRS       0        0         0        1                  31
WALKING_UPSTAIRS         0        0         0       24                  15

Pred                WALKING_UPSTAIRS
True
LAYING                            27
SITTING                            3
STANDING                           1
WALKING                           28
WALKING_DOWNSTAIRS                388
WALKING_UPSTAIRS                  432
```

In [26]:
```
score_48_adam = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 2s 637us/step
```

In [27]:
```
score_48_adam
```

Out[27]:
```
[0.43165912415404384, 0.7648456057007126]
```

LSTM with 64 units

In [28]:
```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(64, 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 [29]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
optimizer='rmsprop',
metrics=['accuracy'])
```

In [30]:
```
# Training the model
history_64_rms = 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 [==============================] - 59s 8ms/step - loss: 1.2178 - acc:
0.4691 - val_loss: 1.0465 - val_acc: 0.5205
Epoch 2/30
7352/7352 [==============================] - 58s 8ms/step - loss: 0.8636 - acc:
0.6321 - val_loss: 0.7916 - val_acc: 0.6362
Epoch 3/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.6813 - acc:
0.7040 - val_loss: 0.6797 - val_acc: 0.7662
Epoch 4/30
7352/7352 [==============================] - 60s 8ms/step - loss: 0.5363 - acc:
0.8069 - val_loss: 0.6028 - val_acc: 0.8113
Epoch 5/30
7352/7352 [==============================] - 59s 8ms/step - loss: 0.4041 - acc:
0.8682 - val_loss: 0.4323 - val_acc: 0.8867
Epoch 6/30
7352/7352 [==============================] - 59s 8ms/step - loss: 0.3008 - acc:
0.9019 - val_loss: 0.5835 - val_acc: 0.8388
Epoch 7/30
7352/7352 [==============================] - 59s 8ms/step - loss: 0.2636 - acc:
0.9159 - val_loss: 0.5152 - val_acc: 0.8517
Epoch 8/30
7352/7352 [==============================] - 60s 8ms/step - loss: 0.2360 - acc:
0.9260 - val_loss: 0.4165 - val_acc: 0.8795
Epoch 9/30
7352/7352 [==============================] - 59s 8ms/step - loss: 0.2117 - acc:
0.9294 - val_loss: 0.4550 - val_acc: 0.8870
Epoch 10/30
7352/7352 [==============================] - 79s 11ms/step - loss: 0.1976 - ac
c: 0.9347 - val_loss: 0.3528 - val_acc: 0.9036
Epoch 11/30
7352/7352 [==============================] - 67s 9ms/step - loss: 0.2377 - acc:
0.9271 - val_loss: 0.4062 - val_acc: 0.8955
Epoch 12/30
7352/7352 [==============================] - 74s 10ms/step - loss: 0.1769 - ac
c: 0.9411 - val_loss: 0.3768 - val_acc: 0.8904
Epoch 13/30
7352/7352 [==============================] - 68s 9ms/step - loss: 0.1923 - acc:
0.9310 - val_loss: 0.4514 - val_acc: 0.8945
Epoch 14/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1765 - acc:
0.9368 - val_loss: 0.5506 - val_acc: 0.8972
Epoch 15/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1685 - acc:
0.9426 - val_loss: 0.3006 - val_acc: 0.9223
Epoch 16/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1678 - acc:
0.9452 - val_loss: 0.2910 - val_acc: 0.9148
Epoch 17/30
```

```
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1462 - acc:
0.9457 - val_loss: 0.3554 - val_acc: 0.9128
Epoch 18/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1557 - acc:
0.9431 - val_loss: 0.2750 - val_acc: 0.9097
Epoch 19/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1496 - acc:
0.9445 - val_loss: 0.2643 - val_acc: 0.9101
Epoch 20/30
7352/7352 [==============================] - 68s 9ms/step - loss: 0.1557 - acc:
0.9456 - val_loss: 0.4241 - val_acc: 0.9138
Epoch 21/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1438 - acc:
0.9489 - val_loss: 0.3581 - val_acc: 0.9182
Epoch 22/30
7352/7352 [==============================] - 66s 9ms/step - loss: 0.1672 - acc:
0.9482 - val_loss: 0.4519 - val_acc: 0.8999
Epoch 23/30
7352/7352 [==============================] - 65s 9ms/step - loss: 0.1646 - acc:
0.9476 - val_loss: 0.3793 - val_acc: 0.8901
Epoch 24/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1342 - acc:
0.9495 - val_loss: 0.4674 - val_acc: 0.9016
Epoch 25/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1487 - acc:
0.9513 - val_loss: 0.3573 - val_acc: 0.9080
Epoch 26/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1430 - acc:
0.9486 - val_loss: 0.4709 - val_acc: 0.8996
Epoch 27/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1375 - acc:
0.9525 - val_loss: 0.3153 - val_acc: 0.9155
Epoch 28/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1383 - acc:
0.9475 - val_loss: 0.3903 - val_acc: 0.9209
Epoch 29/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1412 - acc:
0.9527 - val_loss: 0.4155 - val_acc: 0.9060
Epoch 30/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1524 - acc:
0.9455 - val_loss: 0.8538 - val_acc: 0.8819
```

In [31]:
```
# 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                  3      382       103        1                   0
STANDING                 0       82       447        3                   0
WALKING                  0        0         0      492                   4
WALKING_DOWNSTAIRS       0        0         0       35                 384
WALKING_UPSTAIRS         0        0         0       75                  12

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            2
STANDING                           0
WALKING                            0
WALKING_DOWNSTAIRS                 1
WALKING_UPSTAIRS                 384
```

In [32]:
```
score_64_rms = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 3s 908us/step
```

In [33]:
```
score_64_rms
```

Out[33]:
```
[0.8537755837855637, 0.8819138106549033]
```

LSTM using multilayer with 64 units

In [34]:
```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(64, return_sequences=True, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model.add(Dropout(0.7))
# Configuring the parameters
model.add(LSTM(64, 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_4 (LSTM)                (None, 128, 64)           18944
_________________________________________________________________
dropout_4 (Dropout)          (None, 128, 64)           0
_________________________________________________________________
lstm_5 (LSTM)                (None, 64)                33024
_________________________________________________________________
dropout_5 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_4 (Dense)              (None, 6)                 390
=================================================================
Total params: 52,358
Trainable params: 52,358
Non-trainable params: 0
_________________________________________________________________
```

In [35]:
```
model.compile(loss='categorical_crossentropy',
optimizer='rmsprop',
metrics=['accuracy'])
```

In [36]:
```
# Training the model
history_64_64_rms = 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 [==============================] - 138s 19ms/step - loss: 1.1621 - ac
c: 0.4933 - val_loss: 0.9635 - val_acc: 0.5426
Epoch 2/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.8951 - ac
c: 0.5786 - val_loss: 0.7849 - val_acc: 0.6166
Epoch 3/30
7352/7352 [==============================] - 137s 19ms/step - loss: 0.7275 - ac
c: 0.6386 - val_loss: 0.8098 - val_acc: 0.6149
Epoch 4/30
7352/7352 [==============================] - 141s 19ms/step - loss: 0.7318 - ac
c: 0.6376 - val_loss: 0.8035 - val_acc: 0.6166
Epoch 5/30
7352/7352 [==============================] - 157s 21ms/step - loss: 0.7077 - ac
c: 0.6545 - val_loss: 0.8568 - val_acc: 0.6183
Epoch 6/30
7352/7352 [==============================] - 157s 21ms/step - loss: 0.6574 - ac
c: 0.6787 - val_loss: 1.2631 - val_acc: 0.5497
Epoch 7/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.6084 - ac
c: 0.7247 - val_loss: 0.6831 - val_acc: 0.7828
Epoch 8/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.4588 - ac
c: 0.8436 - val_loss: 0.7575 - val_acc: 0.8073
Epoch 9/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.3437 - ac
c: 0.8999 - val_loss: 0.4098 - val_acc: 0.8748
Epoch 10/30
7352/7352 [==============================] - 133s 18ms/step - loss: 0.2448 - ac
c: 0.9268 - val_loss: 0.3907 - val_acc: 0.8867
Epoch 11/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.2749 - ac
c: 0.9183 - val_loss: 0.4968 - val_acc: 0.8724
Epoch 12/30
7352/7352 [==============================] - 133s 18ms/step - loss: 0.2021 - ac
c: 0.9363 - val_loss: 0.4030 - val_acc: 0.8823
Epoch 13/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.2087 - ac
c: 0.9358 - val_loss: 0.3858 - val_acc: 0.8833
Epoch 14/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1919 - ac
c: 0.9370 - val_loss: 0.4987 - val_acc: 0.8924
Epoch 15/30
7352/7352 [==============================] - 140s 19ms/step - loss: 0.1812 - ac
c: 0.9433 - val_loss: 0.5099 - val_acc: 0.9019
Epoch 16/30
7352/7352 [==============================] - 141s 19ms/step - loss: 0.1769 - ac
c: 0.9448 - val_loss: 0.3621 - val_acc: 0.9114
Epoch 17/30
```

```
7352/7352 [==============================] - 134s 18ms/step - loss: 0.2213 - ac
c: 0.9372 - val_loss: 0.6163 - val_acc: 0.8955
Epoch 18/30
7352/7352 [==============================] - 145s 20ms/step - loss: 0.1690 - ac
c: 0.9441 - val_loss: 0.3735 - val_acc: 0.9036
Epoch 19/30
7352/7352 [==============================] - 137s 19ms/step - loss: 0.1634 - ac
c: 0.9464 - val_loss: 0.3720 - val_acc: 0.9040
Epoch 20/30
7352/7352 [==============================] - 140s 19ms/step - loss: 0.1682 - ac
c: 0.9487 - val_loss: 0.4646 - val_acc: 0.8979
Epoch 21/30
7352/7352 [==============================] - 140s 19ms/step - loss: 0.1546 - ac
c: 0.9482 - val_loss: 0.7664 - val_acc: 0.8877
Epoch 22/30
7352/7352 [==============================] - 138s 19ms/step - loss: 0.1533 - ac
c: 0.9502 - val_loss: 0.3033 - val_acc: 0.9260
Epoch 23/30
7352/7352 [==============================] - 144s 20ms/step - loss: 0.1655 - ac
c: 0.9468 - val_loss: 0.4681 - val_acc: 0.9111
Epoch 24/30
7352/7352 [==============================] - 143s 19ms/step - loss: 0.1563 - ac
c: 0.9480 - val_loss: 0.3611 - val_acc: 0.9111
Epoch 25/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.1467 - ac
c: 0.9470 - val_loss: 0.3619 - val_acc: 0.9284
Epoch 26/30
7352/7352 [==============================] - 137s 19ms/step - loss: 0.1549 - ac
c: 0.9486 - val_loss: 0.3251 - val_acc: 0.8951
Epoch 27/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1458 - ac
c: 0.9453 - val_loss: 0.4668 - val_acc: 0.9284
Epoch 28/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1349 - ac
c: 0.9502 - val_loss: 0.5758 - val_acc: 0.9063
Epoch 29/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1424 - ac
c: 0.9494 - val_loss: 0.3567 - val_acc: 0.9104
Epoch 30/30
7352/7352 [==============================] - 129s 18ms/step - loss: 0.1357 - ac
c: 0.9547 - val_loss: 0.3798 - val_acc: 0.9152
```

In [37]:
```
# Confusion Matrix
print(confusion_matrix(Y_test, model.predict(X_test)))
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 521        0        16        0                   0
SITTING                  0      342       145        0                   0
STANDING                 0       36       496        0                   0
WALKING                  0        0         0      479                  10
WALKING_DOWNSTAIRS       0        0         0        3                 417
WALKING_UPSTAIRS         0        0         0       12                  17

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            4
STANDING                           0
WALKING                            7
WALKING_DOWNSTAIRS                 0
WALKING_UPSTAIRS                 442
```

In [38]:
```
score_64_64_rms = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 6s 2ms/step
```

In [39]:
```
score_64_64_rms
```

Out[39]:
```
[0.3797549853132213, 0.9151679674244995]
```

LSTM with 128 Units

In [55]:
```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(128, input_shape=(timesteps, input_dim)))
model.add(BatchNormalization())
# Adding a dropout layer
model.add(Dropout(0.25))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_11 (LSTM)               (None, 128)               70656
_________________________________________________________________
batch_normalization_1 (Batch (None, 128)               512
_________________________________________________________________
dropout_9 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_7 (Dense)              (None, 6)                 774
=================================================================
Total params: 71,942
Trainable params: 71,686
Non-trainable params: 256
_________________________________________________________________
```

In [56]:
```
model.compile(loss='categorical_crossentropy',
optimizer='rmsprop',
metrics=['accuracy'])
```

In [57]:
```
# Training the model
history_128_rms = 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 [==============================] - 105s 14ms/step - loss: 0.9380 - ac
c: 0.6005 - val_loss: 0.7709 - val_acc: 0.6138
Epoch 2/30
7352/7352 [==============================] - 83s 11ms/step - loss: 0.7620 - ac
c: 0.6370 - val_loss: 0.7600 - val_acc: 0.6325
Epoch 3/30
7352/7352 [==============================] - 85s 12ms/step - loss: 0.5543 - ac
c: 0.7616 - val_loss: 0.4758 - val_acc: 0.8463
Epoch 4/30
7352/7352 [==============================] - 89s 12ms/step - loss: 0.3535 - ac
c: 0.8837 - val_loss: 0.4433 - val_acc: 0.8439
Epoch 5/30
7352/7352 [==============================] - 86s 12ms/step - loss: 0.2282 - ac
c: 0.9202 - val_loss: 0.2980 - val_acc: 0.9097
Epoch 6/30
7352/7352 [==============================] - 89s 12ms/step - loss: 0.1922 - ac
c: 0.9293 - val_loss: 0.2868 - val_acc: 0.9084
Epoch 7/30
7352/7352 [==============================] - 99s 13ms/step - loss: 0.1664 - ac
c: 0.9366 - val_loss: 0.3210 - val_acc: 0.9077
Epoch 8/30
7352/7352 [==============================] - 80s 11ms/step - loss: 0.1723 - ac
c: 0.9381 - val_loss: 0.3590 - val_acc: 0.8904
Epoch 9/30
7352/7352 [==============================] - 80s 11ms/step - loss: 0.1553 - ac
c: 0.9427 - val_loss: 0.4025 - val_acc: 0.9009
Epoch 10/30
7352/7352 [==============================] - 81s 11ms/step - loss: 0.1600 - ac
c: 0.9388 - val_loss: 0.4093 - val_acc: 0.9013
Epoch 11/30
7352/7352 [==============================] - 79s 11ms/step - loss: 0.1442 - ac
c: 0.9411 - val_loss: 0.3144 - val_acc: 0.9087
Epoch 12/30
7352/7352 [==============================] - 80s 11ms/step - loss: 0.1620 - ac
c: 0.9419 - val_loss: 0.4250 - val_acc: 0.8870
Epoch 13/30
7352/7352 [==============================] - 81s 11ms/step - loss: 0.1320 - ac
c: 0.9484 - val_loss: 0.3755 - val_acc: 0.9182
Epoch 14/30
7352/7352 [==============================] - 81s 11ms/step - loss: 0.1419 - ac
c: 0.9478 - val_loss: 0.3352 - val_acc: 0.9237
Epoch 15/30
7352/7352 [==============================] - 81s 11ms/step - loss: 0.1354 - ac
c: 0.9490 - val_loss: 0.5004 - val_acc: 0.9108
Epoch 16/30
7352/7352 [==============================] - 81s 11ms/step - loss: 0.1373 - ac
c: 0.9516 - val_loss: 0.5835 - val_acc: 0.8904
Epoch 17/30
```

```
7352/7352 [==============================] - 82s 11ms/step - loss: 0.1393 - ac
c: 0.9499 - val_loss: 0.3399 - val_acc: 0.9104
Epoch 18/30
7352/7352 [==============================] - 84s 11ms/step - loss: 0.1350 - ac
c: 0.9517 - val_loss: 0.3596 - val_acc: 0.9023
Epoch 19/30
7352/7352 [==============================] - 85s 12ms/step - loss: 0.1467 - ac
c: 0.9474 - val_loss: 0.4728 - val_acc: 0.8880
Epoch 20/30
7352/7352 [==============================] - 86s 12ms/step - loss: 0.1414 - ac
c: 0.9491 - val_loss: 0.5892 - val_acc: 0.8972
Epoch 21/30
7352/7352 [==============================] - 83s 11ms/step - loss: 0.1352 - ac
c: 0.9478 - val_loss: 0.5249 - val_acc: 0.8999
Epoch 22/30
7352/7352 [==============================] - 87s 12ms/step - loss: 0.1519 - ac
c: 0.9449 - val_loss: 0.5121 - val_acc: 0.9094
Epoch 23/30
7352/7352 [==============================] - 101s 14ms/step - loss: 0.1245 - ac
c: 0.9504 - val_loss: 0.4688 - val_acc: 0.9111
Epoch 24/30
7352/7352 [==============================] - 89s 12ms/step - loss: 0.1550 - ac
c: 0.9467 - val_loss: 0.3996 - val_acc: 0.9040
Epoch 25/30
7352/7352 [==============================] - 87s 12ms/step - loss: 0.1342 - ac
c: 0.9505 - val_loss: 0.4258 - val_acc: 0.9158
Epoch 26/30
7352/7352 [==============================] - 88s 12ms/step - loss: 0.1318 - ac
c: 0.9506 - val_loss: 0.4452 - val_acc: 0.9046
Epoch 27/30
7352/7352 [==============================] - 87s 12ms/step - loss: 0.1307 - ac
c: 0.9512 - val_loss: 0.4343 - val_acc: 0.9145
Epoch 28/30
7352/7352 [==============================] - 91s 12ms/step - loss: 0.1301 - ac
c: 0.9516 - val_loss: 0.4638 - val_acc: 0.9060
Epoch 29/30
7352/7352 [==============================] - 82s 11ms/step - loss: 0.1259 - ac
c: 0.9516 - val_loss: 0.5606 - val_acc: 0.9057
Epoch 30/30
7352/7352 [==============================] - 87s 12ms/step - loss: 0.1276 - ac
c: 0.9536 - val_loss: 0.4088 - val_acc: 0.9284
```

In [58]:
```
score_128_rms = model.evaluate(X_test, Y_test)
```

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

In [59]:
```
score_128_rms
```

Out[59]:
```
[0.4087650875253206, 0.9284017645062775]
```

In [48]:
```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(128, return_sequences=True, input_shape=(timesteps, input_dim)))
# model.add(BatchNormalization())
# Adding a dropout layer
model.add(Dropout(0.7))

model.add(LSTM(64))
# 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_9 (LSTM)                (None, 128, 128)          70656
_________________________________________________________________
dropout_7 (Dropout)          (None, 128, 128)          0
_________________________________________________________________
lstm_10 (LSTM)               (None, 64)                49408
_________________________________________________________________
dropout_8 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_6 (Dense)              (None, 6)                 390
=================================================================
Total params: 120,454
Trainable params: 120,454
Non-trainable params: 0
_________________________________________________________________
```

In [49]:
```
model.compile(loss='categorical_crossentropy',
optimizer='rmsprop',
metrics=['accuracy'])
```

In [50]:
```
# Training the model
history_128_64_rms = model.fit(X_train,
Y_train,
batch_size=batch_size,
validation_data=(X_test, Y_test),
epochs=25)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/25
7352/7352 [==============================] - 194s 26ms/step - loss: 1.1063 - ac
c: 0.5137 - val_loss: 0.8138 - val_acc: 0.6447
Epoch 2/25
7352/7352 [==============================] - 171s 23ms/step - loss: 0.7901 - ac
c: 0.6243 - val_loss: 1.3584 - val_acc: 0.4608
Epoch 3/25
7352/7352 [==============================] - 168s 23ms/step - loss: 0.7428 - ac
c: 0.6313 - val_loss: 0.7845 - val_acc: 0.6040
Epoch 4/25
7352/7352 [==============================] - 168s 23ms/step - loss: 0.7310 - ac
c: 0.6421 - val_loss: 0.9128 - val_acc: 0.5993
Epoch 5/25
7352/7352 [==============================] - 170s 23ms/step - loss: 0.8087 - ac
c: 0.6329 - val_loss: 1.2705 - val_acc: 0.4788
Epoch 6/25
7352/7352 [==============================] - 166s 23ms/step - loss: 0.9216 - ac
c: 0.6055 - val_loss: 0.7259 - val_acc: 0.6332
Epoch 7/25
7352/7352 [==============================] - 170s 23ms/step - loss: 0.5462 - ac
c: 0.7735 - val_loss: 0.4805 - val_acc: 0.7933
Epoch 8/25
7352/7352 [==============================] - 183s 25ms/step - loss: 0.3246 - ac
c: 0.8867 - val_loss: 0.5128 - val_acc: 0.8514
Epoch 9/25
7352/7352 [==============================] - 175s 24ms/step - loss: 0.2272 - ac
c: 0.9261 - val_loss: 0.3807 - val_acc: 0.8928
Epoch 10/25
7352/7352 [==============================] - 180s 24ms/step - loss: 0.2003 - ac
c: 0.9298 - val_loss: 0.4115 - val_acc: 0.8979
Epoch 11/25
7352/7352 [==============================] - 168s 23ms/step - loss: 0.2977 - ac
c: 0.8999 - val_loss: 0.7023 - val_acc: 0.7822
Epoch 12/25
7352/7352 [==============================] - 167s 23ms/step - loss: 0.1759 - ac
c: 0.9378 - val_loss: 0.3142 - val_acc: 0.9118
Epoch 13/25
7352/7352 [==============================] - 187s 25ms/step - loss: 0.1855 - ac
c: 0.9353 - val_loss: 0.4181 - val_acc: 0.8951
Epoch 14/25
7352/7352 [==============================] - 188s 26ms/step - loss: 0.1590 - ac
c: 0.9402 - val_loss: 0.3657 - val_acc: 0.9111
Epoch 15/25
7352/7352 [==============================] - 175s 24ms/step - loss: 0.1783 - ac
c: 0.9373 - val_loss: 0.3275 - val_acc: 0.9131
Epoch 16/25
7352/7352 [==============================] - 180s 25ms/step - loss: 0.1426 - ac
c: 0.9468 - val_loss: 0.3577 - val_acc: 0.9135
Epoch 17/25
```

```
7352/7352 [==============================] - 200s 27ms/step - loss: 0.1417 - ac
c: 0.9467 - val_loss: 0.4571 - val_acc: 0.9097
Epoch 18/25
7352/7352 [==============================] - 185s 25ms/step - loss: 0.1476 - ac
c: 0.9472 - val_loss: 0.4135 - val_acc: 0.9033
Epoch 19/25
7352/7352 [==============================] - 179s 24ms/step - loss: 0.1472 - ac
c: 0.9486 - val_loss: 0.4735 - val_acc: 0.8911
Epoch 20/25
7352/7352 [==============================] - 176s 24ms/step - loss: 0.1362 - ac
c: 0.9445 - val_loss: 0.4530 - val_acc: 0.9118
Epoch 21/25
7352/7352 [==============================] - 183s 25ms/step - loss: 0.1450 - ac
c: 0.9457 - val_loss: 0.4248 - val_acc: 0.9087
Epoch 22/25
7352/7352 [==============================] - 177s 24ms/step - loss: 0.1243 - ac
c: 0.9520 - val_loss: 0.3234 - val_acc: 0.9080
Epoch 23/25
7352/7352 [==============================] - 184s 25ms/step - loss: 0.1376 - ac
c: 0.9490 - val_loss: 0.2925 - val_acc: 0.9114
Epoch 24/25
7352/7352 [==============================] - 178s 24ms/step - loss: 0.1435 - ac
c: 0.9474 - val_loss: 0.5265 - val_acc: 0.9019
Epoch 25/25
7352/7352 [==============================] - 175s 24ms/step - loss: 0.1377 - ac
c: 0.9502 - val_loss: 0.3937 - val_acc: 0.9030
```

In [51]:
```
score_128_64_rms = model.evaluate(X_test, Y_test)
```

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

In [52]:
```
score_128_64_rms
```

Out[52]:
```
[0.39368825139877356, 0.9029521547336274]
```

In [ ]:
```

```

In [60]:
```
# Creating table using PrettyTable library
from prettytable import PrettyTable

# Names of models
names =['1 LSTM layer with 32 Units(Optimizer-->rmsprop)','1 LSTM layer with 48 U
'1 LSTM layer with 64 Units(Optimizer-->rmsprop)', '2 LSTM layer with 64 Units(Op
'1 LSTM layer with 128 Units(Optimizer-->rmsprop)', '2 LSTM layer with 128 & 64 U

# Training accuracies
train_acc = [history_32_rms.history['acc'][29], history_48_adam.history['acc'][29
             history_64_64_rms.history['acc'][29], history_128_rms.history['acc'

# Test accuracies
test_acc =[score_32_rms[1], score_48_adam[1], score_64_rms[1], score_64_64_rms[1
numbering = [1,2,3,4, 5, 6]

# Initializing prettytable
ptable = PrettyTable()
# Adding columns
ptable.add_column("S.NO.",numbering)
ptable.add_column("MODEL",names)
ptable.add_column("Training Accuracy",train_acc)
ptable.add_column("Test Accuracy",test_acc)
# Printing the Table
print(ptable)
```

```
+-------+-----------------------------------------------------+-------------
------+--------------------+
| S.NO. |                        MODEL                        | Training Accu
racy  |   Test Accuracy    |
+-------+-----------------------------------------------------+-------------
------+--------------------+
|   1   |    1 LSTM layer with 32 Units(Optimizer-->rmsprop)    | 0.94355277475
51686 | 0.8914149983033594 |
|   2   |      1 LSTM layer with 48 Units(Optimizer--> adam     | 0.83337867247
00761 | 0.7648456057007126 |
|   3   |    1 LSTM layer with 64 Units(Optimizer-->rmsprop)    | 0.94545701849
83678 | 0.8819138106549033 |
|   4   |    2 LSTM layer with 64 Units(Optimizer-->rmsprop)    | 0.95470620239
39065 | 0.9151679674244995 |
|   5   |    1 LSTM layer with 128 Units(Optimizer-->rmsprop)   | 0.95048966267
68227 | 0.9284017645062775 |
|   6   | 2 LSTM layer with 128 & 64 Units(Optimizer-->rmsprop) | 0.95021762785
63657 | 0.9029521547336274 |
+-------+-----------------------------------------------------+-------------
------+--------------------+
```

Conclusion

- Tried multiple architecures and got best 92.84% accuracy using 1 layer architecture of LSTM with 128 Units
- Accuracy is better using RMSProp optimizer compared to Adam