Human Activity Recognition

In [1]:
```
# Importing Libraries
```

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)
```

```
C:\Users\hims1\Anaconda3\lib\site-packages\h5py\__init__.py:36: FutureWarning: Conversion of the second argument of iss
ubdtype from `float` to `np.floating` is deprecated. In future, it will be treated as `np.float64 == np.dtype(float).ty
pe`.
  from ._conv import register_converters as _register_converters
```

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 [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()
```

In [14]:
```
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
```

In [15]:
```
# Initializing parameters
epochs = 30
batch_size = 16
n_hidden = 32
```

- Defining the Architecture of LSTM

In [17]:
```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(n_hidden, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model.add(Dropout(0.5))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

```
WARNING:tensorflow:From C:\Users\hims1\Anaconda3\lib\site-packages\tensorflow\python\framework\op_def_library.py:263: c
olocate_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\hims1\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:3445: calling dr
opout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated 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 [18]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

In [19]:
```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=epochs)
```

```
WARNING:tensorflow:From C:\Users\hims1\Anaconda3\lib\site-packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (fr
om tensorflow.python.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 [==============================] - 30s 4ms/step - loss: 1.3312 - acc: 0.4323 - val_loss: 1.1683 - val_acc: 0.
4846
Epoch 2/30
7352/7352 [==============================] - 27s 4ms/step - loss: 1.0265 - acc: 0.5618 - val_loss: 0.9187 - val_acc: 0.
5945
Epoch 3/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.8139 - acc: 0.6488 - val_loss: 0.7853 - val_acc: 0.
6193
Epoch 4/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.7096 - acc: 0.6689 - val_loss: 0.7408 - val_acc: 0.
6159
Epoch 5/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.6401 - acc: 0.6880 - val_loss: 0.7020 - val_acc: 0.
6664
Epoch 6/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.6230 - acc: 0.6979 - val_loss: 0.7287 - val_acc: 0.
7017
Epoch 7/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.5834 - acc: 0.7262 - val_loss: 0.6355 - val_acc: 0.
7268
Epoch 8/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.5487 - acc: 0.7481 - val_loss: 0.6564 - val_acc: 0.
7306
Epoch 9/30
7352/7352 [==============================] - 27s 4ms/step - loss: 0.4930 - acc: 0.7803 - val_loss: 0.6182 - val_acc: 0.
7360
Epoch 10/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.4556 - acc: 0.7900 - val_loss: 0.5948 - val_acc: 0.
```

```
7112
Epoch 11/30
7352/7352 [==============================] - 27s 4ms/step - loss: 0.4074 - acc: 0.8039 - val_loss: 0.5054 - val_acc: 0.
7421
Epoch 12/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.3793 - acc: 0.8252 - val_loss: 0.4603 - val_acc: 0.
7808
Epoch 13/30
7352/7352 [==============================] - 26s 3ms/step - loss: 0.3665 - acc: 0.8630 - val_loss: 0.5039 - val_acc: 0.
8612
Epoch 14/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.3789 - acc: 0.8876 - val_loss: 0.4645 - val_acc: 0.
8480
Epoch 15/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.3372 - acc: 0.9055 - val_loss: 0.4003 - val_acc: 0.
8687
Epoch 16/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.2673 - acc: 0.9210 - val_loss: 0.4191 - val_acc: 0.
8473
Epoch 17/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.2368 - acc: 0.9236 - val_loss: 0.3641 - val_acc: 0.
8867
Epoch 18/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.2131 - acc: 0.9323 - val_loss: 0.4253 - val_acc: 0.
8833- lo - ETA: 6s - loss: 0.2143  -
Epoch 19/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.2135 - acc: 0.9347 - val_loss: 0.3123 - val_acc: 0.
8985
Epoch 20/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.1983 - acc: 0.9407 - val_loss: 0.4502 - val_acc: 0.
8914
Epoch 21/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.2383 - acc: 0.9308 - val_loss: 0.3462 - val_acc: 0.
8931
Epoch 22/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.1764 - acc: 0.9441 - val_loss: 0.4002 - val_acc: 0.
8958
Epoch 23/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.1920 - acc: 0.9406 - val_loss: 0.5881 - val_acc: 0.
8850lo
Epoch 24/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.2227 - acc: 0.9361 - val_loss: 0.3583 - val_acc: 0.
```

```
8992
Epoch 25/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.1895 - acc: 0.9421 - val_loss: 0.4919 - val_acc: 0.
8748
Epoch 26/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.1905 - acc: 0.9438 - val_loss: 0.3812 - val_acc: 0.
8826
Epoch 27/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.1918 - acc: 0.9416 - val_loss: 0.4873 - val_acc: 0.
8951
Epoch 28/30
7352/7352 [==============================] - 26s 4ms/step - loss: 0.1648 - acc: 0.9463 - val_loss: 0.3695 - val_acc: 0.
8992
Epoch 29/30
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1799 - acc: 0.9442 - val_loss: 0.4771 - val_acc: 0.
8884
Epoch 30/30
7352/7352 [==============================] - 30s 4ms/step - loss: 0.1605 - acc: 0.9465 - val_loss: 0.4689 - val_acc: 0.
8965
```

Out[19]: <keras.callbacks.History at 0x1e6df593eb8>

In [20]:
```
# Confusion Matrix
print(confusion_matrix(Y_test, model.predict(X_test)))
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 510        0         1        0                   0
SITTING                  0      419        51        0                   1
STANDING                 0      132       397        2                   0
WALKING                  0        0         0      466                  29
WALKING_DOWNSTAIRS       0        0         0        1                 416
WALKING_UPSTAIRS         0        1         0       22                  14

Pred                WALKING_UPSTAIRS
True
LAYING                            26
SITTING                           20
STANDING                           1
WALKING                            1
WALKING_DOWNSTAIRS                 3
WALKING_UPSTAIRS                 434
```

In [21]:
```
score = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 1s 415us/step
```

In [22]:
```
score
```

Out[22]:
```
[0.46892408261934193, 0.8965049202578894]
```

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

Assignment - Hyperparameter Tune LSTM model for better accuracy

Trying with different number of LSTM units

Model 1. with 40 LSTM units

In [23]:
```
# Initializing parameters
epochs = 30
batch_size = 16
n_hidden = 40
```

In [24]:
```
# 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, 40)                8000
_________________________________________________________________
dropout_2 (Dropout)          (None, 40)                0
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 246
=================================================================
Total params: 8,246
Trainable params: 8,246
Non-trainable params: 0
_________________________________________________________________
```

In [25]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

In [26]:
```
# 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 [==============================] - 29s 4ms/step - loss: 1.3119 - acc: 0.4222 - val_loss: 1.2055 - val_acc: 0.
4452
Epoch 2/30
7352/7352 [==============================] - 30s 4ms/step - loss: 1.0675 - acc: 0.5169 - val_loss: 1.0054 - val_acc: 0.
5680
Epoch 3/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.9326 - acc: 0.5714 - val_loss: 0.9127 - val_acc: 0.
5653
Epoch 4/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.8286 - acc: 0.6221 - val_loss: 0.9601 - val_acc: 0.
5253
Epoch 5/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.7166 - acc: 0.6884 - val_loss: 0.7821 - val_acc: 0.
6797
Epoch 6/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.5785 - acc: 0.7824 - val_loss: 0.9842 - val_acc: 0.
6922
Epoch 7/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.5031 - acc: 0.8255 - val_loss: 0.6473 - val_acc: 0.
7998
Epoch 8/30
7352/7352 [==============================] - 30s 4ms/step - loss: 0.3928 - acc: 0.8694 - val_loss: 0.5782 - val_acc: 0.
8283
Epoch 9/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.3358 - acc: 0.8891 - val_loss: 0.6768 - val_acc: 0.
8005
Epoch 10/30
7352/7352 [==============================] - 30s 4ms/step - loss: 0.2934 - acc: 0.9072 - val_loss: 0.4664 - val_acc: 0.
8331
Epoch 11/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.2705 - acc: 0.9153 - val_loss: 0.4192 - val_acc: 0.
8585
```

```
Epoch 12/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2342 - acc: 0.9246 - val_loss: 0.4459 - val_acc: 0.
8694
Epoch 13/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2426 - acc: 0.9215 - val_loss: 0.8984 - val_acc: 0.
7669
Epoch 14/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2252 - acc: 0.9300 - val_loss: 0.6818 - val_acc: 0.
8449
Epoch 15/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2385 - acc: 0.9263 - val_loss: 0.6424 - val_acc: 0.
8616
Epoch 16/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2598 - acc: 0.9161 - val_loss: 0.4361 - val_acc: 0.
8602
Epoch 17/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1826 - acc: 0.9344 - val_loss: 0.3396 - val_acc: 0.
8914
Epoch 18/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1868 - acc: 0.9384 - val_loss: 0.3939 - val_acc: 0.
8968
Epoch 19/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2039 - acc: 0.9363 - val_loss: 0.3668 - val_acc: 0.
9074
Epoch 20/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1676 - acc: 0.9414 - val_loss: 0.2577 - val_acc: 0.
9023
Epoch 21/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2006 - acc: 0.9362 - val_loss: 0.3828 - val_acc: 0.
8928
Epoch 22/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1526 - acc: 0.9444 - val_loss: 0.3386 - val_acc: 0.
9046
Epoch 23/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1569 - acc: 0.9464 - val_loss: 0.6617 - val_acc: 0.
8789
Epoch 24/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2104 - acc: 0.9416 - val_loss: 0.7760 - val_acc: 0.
8663
Epoch 25/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1483 - acc: 0.9419 - val_loss: 0.4069 - val_acc: 0.
8951
```

```
Epoch 26/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1469 - acc: 0.9480 - val_loss: 0.5114 - val_acc: 0.
8951
Epoch 27/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1791 - acc: 0.9418 - val_loss: 0.4897 - val_acc: 0.
89961s -
Epoch 28/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1634 - acc: 0.9457 - val_loss: 0.7011 - val_acc: 0.
8911
Epoch 29/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1728 - acc: 0.9429 - val_loss: 0.4513 - val_acc: 0.
8999
Epoch 30/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1565 - acc: 0.9445 - val_loss: 0.3651 - val_acc: 0.
8965
```

Out[26]: <keras.callbacks.History at 0x1e6e3962898>

Model 2. With 64 LSTM Units

In [27]:
```
# Initializing parameters
epochs = 30
batch_size = 16
n_hidden = 64
```

In [28]:
```
# 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 [29]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

In [30]:
```
# 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 [==============================] - 36s 5ms/step - loss: 1.2364 - acc: 0.4732 - val_loss: 1.0499 - val_acc: 0.
5813
Epoch 2/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.9219 - acc: 0.5896 - val_loss: 0.8973 - val_acc: 0.
5986
Epoch 3/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.7434 - acc: 0.6790 - val_loss: 0.7699 - val_acc: 0.
6882
Epoch 4/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.6109 - acc: 0.7408 - val_loss: 0.6081 - val_acc: 0.
7910
Epoch 5/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.4734 - acc: 0.8361 - val_loss: 0.7770 - val_acc: 0.
7808
Epoch 6/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.3605 - acc: 0.8830 - val_loss: 0.5162 - val_acc: 0.
8571
Epoch 7/30
7352/7352 [==============================] - 36s 5ms/step - loss: 0.2978 - acc: 0.9109 - val_loss: 0.4848 - val_acc: 0.
8768
Epoch 8/30
7352/7352 [==============================] - 35s 5ms/step - loss: 0.2614 - acc: 0.9174 - val_loss: 0.3824 - val_acc: 0.
8717
Epoch 9/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.2165 - acc: 0.9270 - val_loss: 0.6313 - val_acc: 0.
8622
Epoch 10/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.2211 - acc: 0.9332 - val_loss: 0.7563 - val_acc: 0.
8575
Epoch 11/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.2020 - acc: 0.9380 - val_loss: 0.7497 - val_acc: 0.
8432
```

```
Epoch 12/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1892 - acc: 0.9380 - val_loss: 0.4028 - val_acc: 0.
8812
Epoch 13/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1882 - acc: 0.9395 - val_loss: 0.3695 - val_acc: 0.
9036
Epoch 14/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1894 - acc: 0.9410 - val_loss: 0.3434 - val_acc: 0.
9043
Epoch 15/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1891 - acc: 0.9425 - val_loss: 0.4316 - val_acc: 0.
8921
Epoch 16/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1717 - acc: 0.9426 - val_loss: 0.3933 - val_acc: 0.
8941
Epoch 17/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1773 - acc: 0.9387 - val_loss: 0.4241 - val_acc: 0.
9002
Epoch 18/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1695 - acc: 0.9457 - val_loss: 0.3248 - val_acc: 0.
9063
Epoch 19/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1703 - acc: 0.9437 - val_loss: 0.3821 - val_acc: 0.
9080
Epoch 20/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1488 - acc: 0.9487 - val_loss: 0.4266 - val_acc: 0.
8965
Epoch 21/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1460 - acc: 0.9476 - val_loss: 0.3937 - val_acc: 0.
9060
Epoch 22/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1448 - acc: 0.9495 - val_loss: 0.5186 - val_acc: 0.
8870
Epoch 23/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1557 - acc: 0.9510 - val_loss: 0.4551 - val_acc: 0.
9019
Epoch 24/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1417 - acc: 0.9471 - val_loss: 0.4141 - val_acc: 0.
9094
Epoch 25/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1469 - acc: 0.9480 - val_loss: 0.4288 - val_acc: 0.
8999
```

```
Epoch 26/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1335 - acc: 0.9514 - val_loss: 0.3602 - val_acc: 0.
9179
Epoch 27/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1405 - acc: 0.9506 - val_loss: 0.3640 - val_acc: 0.
9128
Epoch 28/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1311 - acc: 0.9521 - val_loss: 0.5169 - val_acc: 0.
9026
Epoch 29/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1430 - acc: 0.9476 - val_loss: 0.3377 - val_acc: 0.
9148
Epoch 30/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1300 - acc: 0.9533 - val_loss: 0.4458 - val_acc: 0.
9094
```

Out[30]: <keras.callbacks.History at 0x1e6e95ec4a8>

In [31]:
```
# Confusion Matrix
print(confusion_matrix(Y_test, model.predict(X_test)))
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 511        0        26        0                   0
SITTING                  2      376       111        0                   0
STANDING                 0       67       463        1                   0
WALKING                  0        0         0      456                  15
WALKING_DOWNSTAIRS       0        0         0        0                 418
WALKING_UPSTAIRS         0        0         0        1                  14

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            2
STANDING                           1
WALKING                           25
WALKING_DOWNSTAIRS                 2
WALKING_UPSTAIRS                 456
```

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

```
2947/2947 [==============================] - 2s 599us/step
```

In [33]:
```
score
```

Out[33]:
```
[0.4458080819701784, 0.9093993892093655]
```

3. Model 3 - 128 LSTM units

In [34]:
```
# Initializing parameters
epochs = 30
batch_size = 16
n_hidden = 128
```

In [35]:
```
# 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_4 (LSTM)                (None, 128)               70656     
_________________________________________________________________
dropout_4 (Dropout)          (None, 128)               0         
_________________________________________________________________
dense_4 (Dense)              (None, 6)                 774       
=================================================================
Total params: 71,430
Trainable params: 71,430
Non-trainable params: 0
_________________________________________________________________
```

In [36]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

In [37]:
```
# 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 [==============================] - 63s 9ms/step - loss: 1.2960 - acc: 0.4306 - val_loss: 1.2994 - val_acc:
0.4625
Epoch 2/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.8653 - acc: 0.6260 - val_loss: 0.7847 - val_acc:
0.6502
Epoch 3/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.6582 - acc: 0.7353 - val_loss: 0.6637 - val_acc:
0.7520
Epoch 4/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.5072 - acc: 0.8150 - val_loss: 0.6630 - val_acc:
0.7333
Epoch 5/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.3659 - acc: 0.8791 - val_loss: 0.4133 - val_acc:
0.8585
Epoch 6/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.2462 - acc: 0.9170 - val_loss: 0.3530 - val_acc:
0.8629
Epoch 7/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.2139 - acc: 0.9300 - val_loss: 0.3659 - val_acc:
0.9043
Epoch 8/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1893 - acc: 0.9357 - val_loss: 0.3783 - val_acc:
0.8928
Epoch 9/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1882 - acc: 0.9392 - val_loss: 0.4104 - val_acc:
0.8931
Epoch 10/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1727 - acc: 0.9400 - val_loss: 0.3657 - val_acc:
0.9040
Epoch 11/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1453 - acc: 0.9495 - val_loss: 0.4346 - val_acc:
0.9046
```

```
Epoch 12/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1586 - acc: 0.9440 - val_loss: 0.4312 - val_acc:
0.8918
Epoch 13/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1458 - acc: 0.9475 - val_loss: 0.4586 - val_acc:
0.9036
Epoch 14/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1492 - acc: 0.9472 - val_loss: 0.4406 - val_acc:
0.9135
Epoch 15/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1477 - acc: 0.9476 - val_loss: 0.7404 - val_acc:
0.8856
Epoch 16/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1393 - acc: 0.9513 - val_loss: 0.5410 - val_acc:
0.9023
Epoch 17/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1553 - acc: 0.9480 - val_loss: 0.5414 - val_acc:
0.8955
Epoch 18/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1476 - acc: 0.9514 - val_loss: 0.4589 - val_acc:
0.9033
Epoch 19/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1383 - acc: 0.9505 - val_loss: 0.5552 - val_acc:
0.8914
Epoch 20/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1546 - acc: 0.9426 - val_loss: 0.4748 - val_acc:
0.8897
Epoch 21/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1354 - acc: 0.9490 - val_loss: 0.3471 - val_acc:
0.9182
Epoch 22/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1349 - acc: 0.9506 - val_loss: 0.3687 - val_acc:
0.9125
Epoch 23/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1325 - acc: 0.9525 - val_loss: 0.7346 - val_acc:
0.8904
Epoch 24/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1457 - acc: 0.9520 - val_loss: 0.4050 - val_acc:
0.8996
Epoch 25/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1896 - acc: 0.9361 - val_loss: 0.4694 - val_acc:
0.9026
```

```
Epoch 26/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1558 - acc: 0.9429 - val_loss: 0.3899 - val_acc:
0.9125
Epoch 27/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1324 - acc: 0.9490 - val_loss: 0.4893 - val_acc:
0.9101
Epoch 28/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1270 - acc: 0.9531 - val_loss: 0.4550 - val_acc:
0.8951
Epoch 29/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1351 - acc: 0.9516 - val_loss: 0.4144 - val_acc:
0.9131
Epoch 30/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.2471 - acc: 0.9331 - val_loss: 0.3491 - val_acc:
0.9091
```

Out[37]: <keras.callbacks.History at 0x1e6ed863128>

In [38]:
```
# Confusion Matrix
print(confusion_matrix(Y_test, model.predict(X_test)))
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 534        0         0        0                   0
SITTING                  1      387        98        0                   0
STANDING                 0      102       429        1                   0
WALKING                  0        0         0      441                  27
WALKING_DOWNSTAIRS       0        0         0        0                 418
WALKING_UPSTAIRS         0        1         0        0                   0

Pred                WALKING_UPSTAIRS
True
LAYING                             3
SITTING                            5
STANDING                           0
WALKING                           28
WALKING_DOWNSTAIRS                 2
WALKING_UPSTAIRS                 470
```

In [39]:
```
score = model.evaluate(X_test, Y_test)
```

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

In [40]:
```
score
```

Out[40]:
```
[0.34914769746373947, 0.9090600610790635]
```

Model 4

In [41]:
```
from keras.regularizers import L1L2
from keras.models import load_model
from keras.regularizers import l2
from keras.callbacks import ModelCheckpoint
from keras.layers import LSTM , BatchNormalization
reg = L1L2(0.01, 0.01)
from keras.initializers import he_normal
```

In [42]:
```
model=Sequential()
#neurons=100
model.add(LSTM(100,input_shape=(timesteps,input_dim), kernel_initializer='glorot_normal',
 return_sequences=True, bias_regularizer=reg))
model.add(BatchNormalization())
#dropout =0.6
model.add(Dropout(0.6))
model.add(LSTM(60))
model.add(Dropout(0.6))
model.add(Dense(n_classes,activation='sigmoid'))
#summary
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_5 (LSTM)                (None, 128, 100)          44000
_________________________________________________________________
batch_normalization_1 (Batch (None, 128, 100)          400
_________________________________________________________________
dropout_5 (Dropout)          (None, 128, 100)          0
_________________________________________________________________
lstm_6 (LSTM)                (None, 60)                38640
_________________________________________________________________
dropout_6 (Dropout)          (None, 60)                0
_________________________________________________________________
dense_5 (Dense)              (None, 6)                 366
=================================================================
Total params: 83,406
Trainable params: 83,206
Non-trainable params: 200
_________________________________________________________________
```

In [43]:
```
model.compile(
loss='categorical_crossentropy',
optimizer='rmsprop',
metrics=['accuracy'])
```

In [44]:
```
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 [==============================] - 102s 14ms/step - loss: 2.2795 - acc: 0.6542 - val_loss: 2.0736 - val_acc:
0.6342
Epoch 2/20
7352/7352 [==============================] - 100s 14ms/step - loss: 0.9479 - acc: 0.8498 - val_loss: 0.5123 - val_acc:
0.8599
Epoch 3/20
7352/7352 [==============================] - 100s 14ms/step - loss: 0.3197 - acc: 0.9150 - val_loss: 0.3017 - val_acc:
0.8945
Epoch 4/20
7352/7352 [==============================] - 100s 14ms/step - loss: 0.2457 - acc: 0.9206 - val_loss: 0.3127 - val_acc:
0.8941
Epoch 5/20
7352/7352 [==============================] - 100s 14ms/step - loss: 0.2154 - acc: 0.9323 - val_loss: 0.3894 - val_acc:
0.8558
Epoch 6/20
7352/7352 [==============================] - 100s 14ms/step - loss: 0.1925 - acc: 0.9294 - val_loss: 0.2482 - val_acc:
0.9101
Epoch 7/20
7352/7352 [==============================] - 100s 14ms/step - loss: 0.1825 - acc: 0.9381 - val_loss: 0.3068 - val_acc:
0.9016
Epoch 8/20
7352/7352 [==============================] - 100s 14ms/step - loss: 0.1755 - acc: 0.9389 - val_loss: 0.2350 - val_acc:
0.9152
Epoch 9/20
7352/7352 [==============================] - 100s 14ms/step - loss: 0.1907 - acc: 0.9392 - val_loss: 0.2533 - val_acc:
0.9165
Epoch 10/20
7352/7352 [==============================] - 100s 14ms/step - loss: 0.1718 - acc: 0.9426 - val_loss: 0.3413 - val_acc:
0.9050
Epoch 11/20
7352/7352 [==============================] - 100s 14ms/step - loss: 0.1616 - acc: 0.9404 - val_loss: 0.2424 - val_acc:
0.9223
```

```
Epoch 12/20
7352/7352 [==============================] - 103s 14ms/step - loss: 0.1565 - acc: 0.9412 - val_loss: 0.2935 - val_acc:
0.9091
Epoch 13/20
7352/7352 [==============================] - 104s 14ms/step - loss: 0.1618 - acc: 0.9412 - val_loss: 0.3686 - val_acc:
0.9060
Epoch 14/20
7352/7352 [==============================] - 104s 14ms/step - loss: 0.1778 - acc: 0.9414 - val_loss: 0.4008 - val_acc:
0.9162
Epoch 15/20
7352/7352 [==============================] - 104s 14ms/step - loss: 0.1597 - acc: 0.9436 - val_loss: 0.3874 - val_acc:
0.8850
Epoch 16/20
7352/7352 [==============================] - 102s 14ms/step - loss: 0.1526 - acc: 0.9421 - val_loss: 0.4912 - val_acc:
0.8979
Epoch 17/20
7352/7352 [==============================] - 102s 14ms/step - loss: 0.1505 - acc: 0.9461 - val_loss: 0.3716 - val_acc:
0.9138
Epoch 18/20
7352/7352 [==============================] - 101s 14ms/step - loss: 0.1649 - acc: 0.9438 - val_loss: 0.2851 - val_acc:
0.9179
Epoch 19/20
7352/7352 [==============================] - 101s 14ms/step - loss: 0.1678 - acc: 0.9380 - val_loss: 0.4072 - val_acc:
0.9057
Epoch 20/20
7352/7352 [==============================] - 101s 14ms/step - loss: 0.1507 - acc: 0.9444 - val_loss: 0.3384 - val_acc:
0.9192
```

Out[44]: <keras.callbacks.History at 0x1e6ed85ec18>

In [45]:
```
# 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                  2      443        41        0                   0
STANDING                 0      135       397        0                   0
WALKING                  0        0         0      474                  10
WALKING_DOWNSTAIRS       0        0         0        0                 415
WALKING_UPSTAIRS         0        0         0        4                  24

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            5
STANDING                           0
WALKING                           12
WALKING_DOWNSTAIRS                 5
WALKING_UPSTAIRS                 443
```

In [46]:
```
score = model.evaluate(X_test, Y_test)
```

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

In [47]:
```
score
```

Out[47]:
```
[0.3383917488639054, 0.9192399049881235]
```

Model 5

In [16]:
```
from keras.regularizers import L1L2
from keras.models import load_model
from keras.regularizers import l2
from keras.callbacks import ModelCheckpoint
from keras.layers import LSTM , BatchNormalization
reg = L1L2(0.01, 0.01)
from keras.initializers import he_normal
```

In [17]:
```
model=Sequential()
#neurons=120
model.add(LSTM(150,input_shape=(timesteps,input_dim), kernel_initializer='glorot_normal',
 return_sequences=True, bias_regularizer=reg))
model.add(BatchNormalization())
#dropout =0.7
model.add(Dropout(0.6))
model.add(LSTM(120))
model.add(Dropout(0.6))
model.add(Dense(n_classes,activation='sigmoid'))
#summary
model.summary()
```

```
WARNING:tensorflow:From C:\Users\hims1\Anaconda3\lib\site-packages\tensorflow\python\framework\op_def_library.py:263: c
olocate_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\hims1\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:3445: calling dr
opout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated 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, 128, 150)          96000
_________________________________________________________________
batch_normalization_1 (Batch (None, 128, 150)          600
_________________________________________________________________
dropout_1 (Dropout)          (None, 128, 150)          0
_________________________________________________________________
lstm_2 (LSTM)                (None, 120)               130080
_________________________________________________________________
dropout_2 (Dropout)          (None, 120)               0
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 726
=================================================================
Total params: 227,406
Trainable params: 227,106
Non-trainable params: 300
_________________________________________________________________
```

In [18]:
```
#https://www.tensorflow.org/tensorboard/scalars_and_keras
filepath="weights.best.hdf5"
from keras.callbacks import ModelCheckpoint, EarlyStopping, TensorBoard
checkpoint_1 = ModelCheckpoint(filepath,
                               monitor="val_acc",
                               mode="max",
                               save_best_only=True,
                               verbose=1)
tensorboard_1 = TensorBoard(log_dir='graph_one', batch_size=16,update_freq='epoch')
callbacks_1 = [checkpoint_1,tensorboard_1]
```

In [19]:
```
model.compile(
loss='categorical_crossentropy',
optimizer='rmsprop',
metrics=['accuracy'])
```

In [20]:
```
hist=model.fit(X_train,
Y_train,
batch_size=batch_size,
validation_data=(X_test, Y_test),
epochs=20,
callbacks=callbacks_1)
```

```
Epoch 17/20
7352/7352 [==============================] - 182s 25ms/step - loss: 0.1535 - acc: 0.9464 - val_loss: 0.6074 - val_ac
c: 0.8795

Epoch 00017: val_acc did not improve from 0.93247
Epoch 18/20
7352/7352 [==============================] - 181s 25ms/step - loss: 0.1456 - acc: 0.9470 - val_loss: 0.5414 - val_ac
c: 0.9009

Epoch 00018: val_acc did not improve from 0.93247
Epoch 19/20
7352/7352 [==============================] - 182s 25ms/step - loss: 0.1579 - acc: 0.9410 - val_loss: 0.3574 - val_ac
c: 0.9192

Epoch 00019: val_acc did not improve from 0.93247
Epoch 20/20
7352/7352 [==============================] - 183s 25ms/step - loss: 0.1613 - acc: 0.9422 - val_loss: 0.3894 - val_ac
c: 0.9087

Epoch 00020: val_acc did not improve from 0.93247
```

In [21]:
```
# Confusion Matrix
print(confusion_matrix(Y_test, model.predict(X_test)))
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 537        0         0        0                   0
SITTING                  4      401        84        0                   0
STANDING                 0      116       416        0                   0
WALKING                  0        1         0      465                  23
WALKING_DOWNSTAIRS       0        2         0        0                 415
WALKING_UPSTAIRS         0       19         0        3                   5

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            2
STANDING                           0
WALKING                            7
WALKING_DOWNSTAIRS                 3
WALKING_UPSTAIRS                 444
```

In [25]:
```
model=Sequential()
#neurons=120
model.add(LSTM(150,input_shape=(timesteps,input_dim), kernel_initializer='glorot_normal',
 return_sequences=True, bias_regularizer=reg))
model.add(BatchNormalization())
#dropout =0.7
model.add(Dropout(0.6))
model.add(LSTM(120))
model.add(Dropout(0.6))
model.add(Dense(n_classes,activation='sigmoid'))
#summary
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_3 (LSTM)                (None, 128, 150)          96000
_________________________________________________________________
batch_normalization_2 (Batch (None, 128, 150)          600
_________________________________________________________________
dropout_3 (Dropout)          (None, 128, 150)          0
_________________________________________________________________
lstm_4 (LSTM)                (None, 120)               130080
_________________________________________________________________
dropout_4 (Dropout)          (None, 120)               0
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 726
=================================================================
Total params: 227,406
Trainable params: 227,106
Non-trainable params: 300
_________________________________________________________________
```

In [26]:
```
model.load_weights("weights.best.hdf5")
```

In [27]:
```
model.compile(
loss='categorical_crossentropy',
optimizer='rmsprop',
metrics=['accuracy'])
```

In [28]:
```
score = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 12s 4ms/step
```

In [29]:
```
score
```

Out[29]:
```
[0.36393380713630763, 0.9324737020699015]
```

Comparison of Models

In [58]:
```
from prettytable import PrettyTable
x = PrettyTable()
x.field_names = ["Sr. No.","Model Name", "Test Accuracy", "Test loss"]
x.add_row(["1","1 layer with 40 LSTM Units","0.90", "0.30"])
x.add_row(["2","1 layer with 64 LSTM units","0.91", "0.39"])
x.add_row(["3","1 layer with 128 LSTM units","0.92", "0.47"])
x.add_row(["4","2 layers with 100 & 60 LSTM units","0.92", "0.30"])
x.add_row(["5","2 layers with 150 & 120 LSTM units","0.93", "0.36"])
print(x)
```

```
+---------+-------------------------------------+---------------+-----------+
| Sr. No. |              Model Name             | Test Accuracy | Test loss |
+---------+-------------------------------------+---------------+-----------+
|    1    |     1 layer with 40 LSTM Units      |      0.90     |    0.30   |
|    2    |     1 layer with 64 LSTM units      |      0.91     |    0.39   |
|    3    |    1 layer with 128 LSTM units      |      0.92     |    0.47   |
|    4    | 2 layers with 100 & 60 LSTM units   |      0.92     |    0.30   |
|    5    | 2 layers with 150 & 120 LSTM units  |      0.93     |    0.36   |
+---------+-------------------------------------+---------------+-----------+
```

1. We were able to acheive best accuracy with model 5 which has 2 layers of LSTM with 150 & 120 units.
2. We got a Test accuracy of 93.25% & Test loss of 0.36 using the above model.

In [ ]: