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

In [2]:
```
import pandas as pd
import numpy as np
```

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

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

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

Data

In [4]:
```
# Data directory
DATADIR = 'UCI_HAR_Dataset'
```

In [5]:
```
# Raw data signals
# Signals are from Accelerometer and Gyroscope
# The signals are in x,y,z directions
# Sensor signals are filtered to have only body acceleration
# excluding the acceleration due to gravity
# Triaxial acceleration from the accelerometer is total acceleration
SIGNALS = [
    "body_acc_x",
    "body_acc_y",
    "body_acc_z",
    "body_gyro_x",
    "body_gyro_y",
    "body_gyro_z",
    "total_acc_x",
    "total_acc_y",
    "total_acc_z"
]
```

In [6]:
```
# Utility function to read the data from csv file
def _read_csv(filename):
    return pd.read_csv(filename, delim_whitespace=True, header=None)

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

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

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

In [7]:
```
def load_y(subset):
    """
    The objective that we are trying to predict is a integer, from 1 to 6,
    that represents a human activity. We return a binary representation of
    every sample objective as a 6 bits vector using One Hot Encoding
    (https://pandas.pydata.org/pandas-docs/stable/generated/pandas.get_dummies.html)
    """
    filename = f'UCI_HAR_Dataset/{subset}/y_{subset}.txt'
    y = _read_csv(filename)[0]

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

In [8]:
```
def load_data():
    """
    Obtain the dataset from multiple files.
    Returns: X_train, X_test, y_train, y_test
    """
    X_train, X_test = load_signals('train'), load_signals('test')
    y_train, y_test = load_y('train'), load_y('test')

    return X_train, X_test, y_train, y_test
```

In [9]:
```
# Importing tensorflow
np.random.seed(42)
import tensorflow as tf
tf.set_random_seed(42)
```

In [10]:
```
# Configuring a session
session_conf = tf.ConfigProto(
    intra_op_parallelism_threads=1,
    inter_op_parallelism_threads=1
)
```

In [11]:
```
# Import Keras
from keras import backend as K
sess = tf.Session(graph=tf.get_default_graph(), config=session_conf)
K.set_session(sess)
```

In [12]:
```
# Importing libraries
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers.core import Dense, Dropout
```

In [13]:
```
# Initializing parameters
epochs = 30
batch_size = 16
n_hidden = 32
```

In [14]:
```
# Utility function to count the number of classes
def _count_classes(y):
    return len(set([tuple(category) for category in y]))
```

In [15]:
```
# Loading the train and test data
X_train, X_test, Y_train, Y_test = load_data()
```

```
C:\Users\LENOVO\Anaconda3\lib\site-packages\ipykernel_launcher.py:12: FutureW
arning: Method .as_matrix will be removed in a future version. Use .values in
stead.
  if sys.path[0] == '':
```

In [16]:
```
len(X_train[0][0])
```

Out[16]:
```
9
```

In [17]:
```
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 [26]:
```
# 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\LENOVO\Anaconda3\lib\site-packages\keras\bac
kend\tensorflow_backend.py:541: The name tf.placeholder is deprecated. Please
use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\bac
kend\tensorflow_backend.py:4432: The name tf.random_uniform is deprecated. Pl
ease use tf.random.uniform instead.

WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\bac
kend\tensorflow_backend.py:148: The name tf.placeholder_with_default is depre
cated. Please use tf.compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\bac
kend\tensorflow_backend.py:3733: calling dropout (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 - k
eep_prob`.
Model: "sequential_2"
_________________________________________________________________
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 [27]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

```
WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\opt
imizers.py:793: The name tf.train.Optimizer is deprecated. Please use tf.comp
at.v1.train.Optimizer instead.

WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\bac
kend\tensorflow_backend.py:3576: The name tf.log is deprecated. Please use t
f.math.log instead.
```

In [1]:
```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=epochs)
```

In [24]:
```
# Confusion Matrix
print(confusion_matrix(Y_test, model.predict(X_test)))
```

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

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

In [27]:
```
score = model.evaluate(X_test, Y_test)
```

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

In [28]:
```
score
```

Out[28]:
```
[0.3087582236972612, 0.9097387173396675]
```

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

Assignment:

Model-1:

In [17]:
```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(40, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model.add(Dropout(0.5))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

```
WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\bac
kend\tensorflow_backend.py:66: The name tf.get_default_graph is deprecated. P
lease use tf.compat.v1.get_default_graph instead.

WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\bac
kend\tensorflow_backend.py:541: The name tf.placeholder is deprecated. Please
use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\bac
kend\tensorflow_backend.py:4432: The name tf.random_uniform is deprecated. Pl
ease use tf.random.uniform instead.

WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\bac
kend\tensorflow_backend.py:148: The name tf.placeholder_with_default is depre
cated. Please use tf.compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\bac
kend\tensorflow_backend.py:3733: calling dropout (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 - k
eep_prob`.
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 40)                8000
_________________________________________________________________
dropout_1 (Dropout)          (None, 40)                0
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 246
=================================================================
Total params: 8,246
Trainable params: 8,246
Non-trainable params: 0
_________________________________________________________________
```

In [18]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

```
WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\opt
imizers.py:793: The name tf.train.Optimizer is deprecated. Please use tf.comp
at.v1.train.Optimizer instead.

WARNING:tensorflow:From C:\Users\LENOVO\Anaconda3\lib\site-packages\keras\bac
kend\tensorflow_backend.py:3576: The name tf.log is deprecated. Please use t
f.math.log instead.
```

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\LENOVO\AppData\Roaming\Python\Python36\site-
packages\tensorflow\python\ops\math_grad.py:1250: add_dispatch_support.<local
s>.wrapper (from tensorflow.python.ops.array_ops) is deprecated and will be r
emoved in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 123s 17ms/step - loss: 1.2862 -
acc: 0.4539 - val_loss: 1.1677 - val_acc: 0.5059
Epoch 2/30
7352/7352 [==============================] - 121s 16ms/step - loss: 1.1326 -
acc: 0.5135 - val_loss: 1.0953 - val_acc: 0.5161
Epoch 3/30
7352/7352 [==============================] - 118s 16ms/step - loss: 0.9043 -
acc: 0.6050 - val_loss: 0.9573 - val_acc: 0.6050
Epoch 4/30
7352/7352 [==============================] - 124s 17ms/step - loss: 0.7925 -
acc: 0.6255 - val_loss: 1.1005 - val_acc: 0.5249
Epoch 5/30
7352/7352 [==============================] - 131s 18ms/step - loss: 0.7313 -
acc: 0.6461 - val_loss: 0.8016 - val_acc: 0.6237
Epoch 6/30
7352/7352 [==============================] - 132s 18ms/step - loss: 0.7000 -
acc: 0.6665 - val_loss: 0.9460 - val_acc: 0.5989
Epoch 7/30
7352/7352 [==============================] - 131s 18ms/step - loss: 0.7608 -
acc: 0.6526 - val_loss: 0.7669 - val_acc: 0.6410
Epoch 8/30
7352/7352 [==============================] - 131s 18ms/step - loss: 0.6546 -
acc: 0.7021 - val_loss: 0.7351 - val_acc: 0.6919
Epoch 9/30
7352/7352 [==============================] - 132s 18ms/step - loss: 0.6057 -
acc: 0.7602 - val_loss: 0.7004 - val_acc: 0.7750
Epoch 10/30
7352/7352 [==============================] - 144s 20ms/step - loss: 0.5372 -
acc: 0.7988 - val_loss: 0.8634 - val_acc: 0.7027
Epoch 11/30
7352/7352 [==============================] - 139s 19ms/step - loss: 0.4524 -
acc: 0.8482 - val_loss: 0.6696 - val_acc: 0.8154
Epoch 12/30
7352/7352 [==============================] - 138s 19ms/step - loss: 0.3766 -
acc: 0.8821 - val_loss: 0.6002 - val_acc: 0.8402
Epoch 13/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.3165 -
acc: 0.9036 - val_loss: 0.5824 - val_acc: 0.8422
Epoch 14/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.2830 -
acc: 0.9097 - val_loss: 0.5283 - val_acc: 0.8514
Epoch 15/30
7352/7352 [==============================] - 138s 19ms/step - loss: 0.2437 -
acc: 0.9196 - val_loss: 0.5322 - val_acc: 0.8643
Epoch 16/30
7352/7352 [==============================] - 144s 20ms/step - loss: 0.2347 -
acc: 0.9263 - val_loss: 0.4037 - val_acc: 0.8870
Epoch 17/30
7352/7352 [==============================] - 137s 19ms/step - loss: 0.2268 -
```

```
acc: 0.9280 - val_loss: 0.5046 - val_acc: 0.8819
Epoch 18/30
7352/7352 [==============================] - 141s 19ms/step - loss: 0.1989 -
acc: 0.9358 - val_loss: 0.5478 - val_acc: 0.8897
Epoch 19/30
7352/7352 [==============================] - 136s 19ms/step - loss: 0.1892 -
acc: 0.9361 - val_loss: 0.4753 - val_acc: 0.8850
Epoch 20/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1910 -
acc: 0.9372 - val_loss: 0.5522 - val_acc: 0.8812
Epoch 21/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.2002 -
acc: 0.9354 - val_loss: 0.6198 - val_acc: 0.8772
Epoch 22/30
7352/7352 [==============================] - 136s 19ms/step - loss: 0.1783 -
acc: 0.9387 - val_loss: 0.5255 - val_acc: 0.8839
Epoch 23/30
7352/7352 [==============================] - 136s 19ms/step - loss: 0.1737 -
acc: 0.9361 - val_loss: 0.5526 - val_acc: 0.8877
Epoch 24/30
7352/7352 [==============================] - 137s 19ms/step - loss: 0.1692 -
acc: 0.9410 - val_loss: 0.9415 - val_acc: 0.6393
Epoch 25/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1814 -
acc: 0.9361 - val_loss: 0.6644 - val_acc: 0.8731
Epoch 26/30
7352/7352 [==============================] - 136s 19ms/step - loss: 0.1997 -
acc: 0.9344 - val_loss: 0.5646 - val_acc: 0.8890
Epoch 27/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.1624 -
acc: 0.9459 - val_loss: 0.4422 - val_acc: 0.8931
Epoch 28/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.1621 -
acc: 0.9449 - val_loss: 0.4937 - val_acc: 0.8850
Epoch 29/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.1584 -
acc: 0.9470 - val_loss: 0.5160 - val_acc: 0.8992
Epoch 30/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.1656 -
acc: 0.9440 - val_loss: 0.5032 - val_acc: 0.8894
```

Out[19]: <keras.callbacks.History at 0x1355507ed30>

In [20]:
```
# 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      405        62        0                   5
STANDING                 0      106       424        1                   0
WALKING                  0        0         7      406                  27
WALKING_DOWNSTAIRS       0        0         0        0                 417
WALKING_UPSTAIRS         0        0         0        1                  11

Pred                WALKING_UPSTAIRS
True
LAYING                            27
SITTING                           19
STANDING                           1
WALKING                           56
WALKING_DOWNSTAIRS                 3
WALKING_UPSTAIRS                 459
```

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

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

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

Out[22]:
```
[0.5031543293462738, 0.8893790295215473]
```

In [ ]:
```

```

Model-2:

In [47]:
```
# code from https://keras.io/regularizers/
from keras.regularizers import L1L2
from keras.models import load_model
from keras.callbacks import ModelCheckpoint
from keras.layers import LSTM , BatchNormalization
reg = L1L2(0.01, 0.01)
```

In [42]:
```
model = Sequential()
model.add(LSTM(100, input_shape=(timesteps, input_dim), kernel_initializer='gl
orot_normal' , return_sequences=True, bias_regularizer=reg))
model.add(BatchNormalization())
model.add(Dropout(0.70))
model.add(LSTM(50))
model.add(Dropout(0.70))
model.add(Dense(n_classes, activation='sigmoid'))
print("Model Summary: ")
model.summary()
```

```
Model Summary:
Model: "sequential_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_8 (LSTM)                (None, 128, 100)          44000
_________________________________________________________________
batch_normalization_4 (Batch (None, 128, 100)          400
_________________________________________________________________
dropout_7 (Dropout)          (None, 128, 100)          0
_________________________________________________________________
lstm_9 (LSTM)                (None, 50)                30200
_________________________________________________________________
dropout_8 (Dropout)          (None, 50)                0
_________________________________________________________________
dense_4 (Dense)              (None, 6)                 306
=================================================================
Total params: 74,906
Trainable params: 74,706
Non-trainable params: 200
_________________________________________________________________
```

In [43]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
checkpoint_3 = ModelCheckpoint("model_6.h5",monitor="val_acc",mode="max",save_
best_only = True,verbose=1)
```

In [44]:
```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=20,callbacks=[checkpoint_3])
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/20
7352/7352 [==============================] - 248s 34ms/step - loss: 2.3953 -
acc: 0.5909 - val_loss: 1.8404 - val_acc: 0.5999

Epoch 00001: val_acc improved from -inf to 0.59993, saving model to model_6.h
5
Epoch 2/20
7352/7352 [==============================] - 235s 32ms/step - loss: 1.1531 -
acc: 0.6884 - val_loss: 1.0318 - val_acc: 0.5999

Epoch 00002: val_acc did not improve from 0.59993
Epoch 3/20
7352/7352 [==============================] - 232s 32ms/step - loss: 0.5914 -
acc: 0.7877 - val_loss: 0.4268 - val_acc: 0.8660

Epoch 00003: val_acc improved from 0.59993 to 0.86597, saving model to model_
6.h5
Epoch 4/20
7352/7352 [==============================] - 232s 32ms/step - loss: 0.4100 -
acc: 0.8750 - val_loss: 0.3189 - val_acc: 0.8968

Epoch 00004: val_acc improved from 0.86597 to 0.89684, saving model to model_
6.h5
Epoch 5/20
7352/7352 [==============================] - 234s 32ms/step - loss: 0.3255 -
acc: 0.9000 - val_loss: 0.3291 - val_acc: 0.8931

Epoch 00005: val_acc did not improve from 0.89684
Epoch 6/20
7352/7352 [==============================] - 237s 32ms/step - loss: 0.2785 -
acc: 0.9135 - val_loss: 0.3620 - val_acc: 0.8850

Epoch 00006: val_acc did not improve from 0.89684
Epoch 7/20
7352/7352 [==============================] - 245s 33ms/step - loss: 0.2566 -
acc: 0.9221 - val_loss: 0.3146 - val_acc: 0.8921

Epoch 00007: val_acc did not improve from 0.89684
Epoch 8/20
7352/7352 [==============================] - 237s 32ms/step - loss: 0.2339 -
acc: 0.9280 - val_loss: 0.5996 - val_acc: 0.8728

Epoch 00008: val_acc did not improve from 0.89684
Epoch 9/20
7352/7352 [==============================] - 236s 32ms/step - loss: 0.2438 -
acc: 0.9244 - val_loss: 0.2507 - val_acc: 0.9209

Epoch 00009: val_acc improved from 0.89684 to 0.92094, saving model to model_
6.h5
Epoch 10/20
7352/7352 [==============================] - 236s 32ms/step - loss: 0.2304 -
acc: 0.9256 - val_loss: 0.3254 - val_acc: 0.8979

Epoch 00010: val_acc did not improve from 0.92094
Epoch 11/20
7352/7352 [==============================] - 235s 32ms/step - loss: 0.2392 -
```

```
acc: 0.9244 - val_loss: 0.4724 - val_acc: 0.8558

Epoch 00011: val_acc did not improve from 0.92094
Epoch 12/20
7352/7352 [==============================] - 242s 33ms/step - loss: 0.2030 -
acc: 0.9314 - val_loss: 0.3208 - val_acc: 0.9186

Epoch 00012: val_acc did not improve from 0.92094
Epoch 13/20
7352/7352 [==============================] - 246s 33ms/step - loss: 0.2041 -
acc: 0.9369 - val_loss: 0.2538 - val_acc: 0.9233

Epoch 00013: val_acc improved from 0.92094 to 0.92331, saving model to model_
6.h5
Epoch 14/20
7352/7352 [==============================] - 236s 32ms/step - loss: 0.2126 -
acc: 0.9310 - val_loss: 0.2945 - val_acc: 0.8951

Epoch 00014: val_acc did not improve from 0.92331
Epoch 15/20
7352/7352 [==============================] - 237s 32ms/step - loss: 0.2042 -
acc: 0.9306 - val_loss: 0.2545 - val_acc: 0.9338

Epoch 00015: val_acc improved from 0.92331 to 0.93383, saving model to model_
6.h5
Epoch 16/20
7352/7352 [==============================] - 235s 32ms/step - loss: 0.2000 -
acc: 0.9339 - val_loss: 0.2633 - val_acc: 0.9179

Epoch 00016: val_acc did not improve from 0.93383
Epoch 17/20
7352/7352 [==============================] - 235s 32ms/step - loss: 0.1971 -
acc: 0.9334 - val_loss: 0.3233 - val_acc: 0.9203

Epoch 00017: val_acc did not improve from 0.93383
Epoch 18/20
7352/7352 [==============================] - 235s 32ms/step - loss: nan - ac
c: 0.3274 - val_loss: nan - val_acc: 0.1683

Epoch 00018: val_acc did not improve from 0.93383
Epoch 19/20
7352/7352 [==============================] - 235s 32ms/step - loss: nan - ac
c: 0.1668 - val_loss: nan - val_acc: 0.1683

Epoch 00019: val_acc did not improve from 0.93383
Epoch 20/20
7352/7352 [==============================] - 235s 32ms/step - loss: nan - ac
c: 0.1668 - val_loss: nan - val_acc: 0.1683

Epoch 00020: val_acc did not improve from 0.93383
```

Out[44]: <keras.callbacks.History at 0x15ca245cf28>

In [48]:
```
model = load_model('model_6.h5')
```

In [49]:
```
# 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      389        98        0                   0
STANDING                 0       47       482        3                   0
WALKING                  0        0         0      475                  20
WALKING_DOWNSTAIRS       0        0         0        0                 420
WALKING_UPSTAIRS         0        0         0        8                  14

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            4
STANDING                           0
WALKING                            1
WALKING_DOWNSTAIRS                 0
WALKING_UPSTAIRS                 449
```

In [50]:
```
score = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 17s 6ms/step
```

In [51]:
```
score
```

Out[51]:
```
[0.2544564464753497, 0.9338310145911096]
```

Model-3:

In [32]:
```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(70, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model.add(Dropout(0.7))
# Adding a dense output layer with sigmoid activation
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()
```

```
WARNING:tensorflow:Large dropout rate: 0.7 (>0.5). In TensorFlow 2.x, dropout
() uses dropout rate instead of keep_prob. Please ensure that this is intende
d.
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_4 (LSTM)                (None, 70)                22400
_________________________________________________________________
dropout_4 (Dropout)          (None, 70)                0
_________________________________________________________________
dense_4 (Dense)              (None, 6)                 426
=================================================================
Total params: 22,826
Trainable params: 22,826
Non-trainable params: 0
_________________________________________________________________
```

In [33]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

In [34]:
```
# 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 [==============================] - 126s 17ms/step - loss: 1.2871 -
acc: 0.4425 - val_loss: 1.2365 - val_acc: 0.4669
Epoch 2/30
7352/7352 [==============================] - 137s 19ms/step - loss: 1.0508 -
acc: 0.5423 - val_loss: 1.0076 - val_acc: 0.6088
Epoch 3/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.8036 -
acc: 0.6270 - val_loss: 0.8368 - val_acc: 0.5979
Epoch 4/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.7237 -
acc: 0.6608 - val_loss: 0.7729 - val_acc: 0.6159
Epoch 5/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.9305 -
acc: 0.6065 - val_loss: 0.8180 - val_acc: 0.5826
Epoch 6/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.6588 -
acc: 0.7073 - val_loss: 0.6039 - val_acc: 0.7238
Epoch 7/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.5586 -
acc: 0.7560 - val_loss: 0.5590 - val_acc: 0.7631
Epoch 8/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.5349 -
acc: 0.7851 - val_loss: 0.4933 - val_acc: 0.8368
Epoch 9/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.5117 -
acc: 0.8266 - val_loss: 0.4738 - val_acc: 0.8269
Epoch 10/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.3452 -
acc: 0.8925 - val_loss: 0.3833 - val_acc: 0.8663
Epoch 11/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.2779 -
acc: 0.9172 - val_loss: 0.3797 - val_acc: 0.8856
Epoch 12/30
7352/7352 [==============================] - 141s 19ms/step - loss: 0.2701 -
acc: 0.9221 - val_loss: 0.3169 - val_acc: 0.8711
Epoch 13/30
7352/7352 [==============================] - 138s 19ms/step - loss: 0.2534 -
acc: 0.9286 - val_loss: 0.2987 - val_acc: 0.9009
Epoch 14/30
7352/7352 [==============================] - 132s 18ms/step - loss: 0.2325 -
acc: 0.9309 - val_loss: 0.3455 - val_acc: 0.8951
Epoch 15/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.2166 -
acc: 0.9340 - val_loss: 0.2628 - val_acc: 0.9091
Epoch 16/30
7352/7352 [==============================] - 129s 18ms/step - loss: 0.2118 -
acc: 0.9374 - val_loss: 0.4613 - val_acc: 0.8860
Epoch 17/30
7352/7352 [==============================] - 129s 18ms/step - loss: 0.2248 -
acc: 0.9343 - val_loss: 0.5067 - val_acc: 0.8755
Epoch 18/30
7352/7352 [==============================] - 129s 17ms/step - loss: 0.2107 -
acc: 0.9347 - val_loss: 0.2522 - val_acc: 0.9230
Epoch 19/30
7352/7352 [==============================] - 128s 17ms/step - loss: 0.2013 -
```

```
acc: 0.9433 - val_loss: 0.2784 - val_acc: 0.9175
Epoch 20/30
7352/7352 [==============================] - 151s 21ms/step - loss: 0.1917 -
acc: 0.9402 - val_loss: 0.2704 - val_acc: 0.9230
Epoch 21/30
7352/7352 [==============================] - 137s 19ms/step - loss: 0.1846 -
acc: 0.9450 - val_loss: 0.2450 - val_acc: 0.9186
Epoch 22/30
7352/7352 [==============================] - 130s 18ms/step - loss: 0.1932 -
acc: 0.9450 - val_loss: 0.2845 - val_acc: 0.9060
Epoch 23/30
7352/7352 [==============================] - 129s 18ms/step - loss: 0.2236 -
acc: 0.9358 - val_loss: 0.3241 - val_acc: 0.9158
Epoch 24/30
7352/7352 [==============================] - 130s 18ms/step - loss: 0.1734 -
acc: 0.9429 - val_loss: 0.2799 - val_acc: 0.9162
Epoch 25/30
7352/7352 [==============================] - 128s 17ms/step - loss: 0.1593 -
acc: 0.9456 - val_loss: 0.3036 - val_acc: 0.9186
Epoch 26/30
7352/7352 [==============================] - 128s 17ms/step - loss: 0.3904 -
acc: 0.9135 - val_loss: 0.3578 - val_acc: 0.9063
Epoch 27/30
7352/7352 [==============================] - 128s 17ms/step - loss: 0.1722 -
acc: 0.9430 - val_loss: 0.6189 - val_acc: 0.8880
Epoch 28/30
7352/7352 [==============================] - 132s 18ms/step - loss: 0.2481 -
acc: 0.9327 - val_loss: 0.4034 - val_acc: 0.9030
Epoch 29/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1677 -
acc: 0.9479 - val_loss: 0.4361 - val_acc: 0.9131
Epoch 30/30
7352/7352 [==============================] - 130s 18ms/step - loss: 0.3103 -
acc: 0.9259 - val_loss: 0.4625 - val_acc: 0.8989
```

Out[34]: <keras.callbacks.History at 0x1361bad9630>

In [35]:
```
# Confusion Matrix
print(confusion_matrix(Y_test, model.predict(X_test)))
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 485       13        27        1                   9
SITTING                  0      373       111        4                   0
STANDING                 0       81       450        0                   0
WALKING                  0        0         0      462                  19
WALKING_DOWNSTAIRS       0        0         0        1                 416
WALKING_UPSTAIRS         0        0         0        2                   6

Pred                WALKING_UPSTAIRS
True
LAYING                             2
SITTING                            3
STANDING                           1
WALKING                           15
WALKING_DOWNSTAIRS                 3
WALKING_UPSTAIRS                 463
```

In [36]:
```
score = model.evaluate(X_test, Y_test)
```

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

In [37]:
```
score
```

Out[37]:
```
[0.46246355840847214, 0.8988802171700034]
```

In [ ]:

In [ ]:

Model-4:

In [23]:
```
# Initiliazing the sequential model
model = Sequential()
# Configuring the parameters
model.add(LSTM(50, 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()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_2 (LSTM)                (None, 50)                12000
_________________________________________________________________
dropout_2 (Dropout)          (None, 50)                0
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 306
=================================================================
Total params: 12,306
Trainable params: 12,306
Non-trainable params: 0
_________________________________________________________________
```

In [24]:
```
# Compiling the model
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

In [25]:
```
# 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 [==============================] - 125s 17ms/step - loss: 1.1955 -
acc: 0.4834 - val_loss: 0.8875 - val_acc: 0.6159
Epoch 2/30
7352/7352 [==============================] - 125s 17ms/step - loss: 0.8413 -
acc: 0.6178 - val_loss: 0.7971 - val_acc: 0.6057
Epoch 3/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.7409 -
acc: 0.6605 - val_loss: 0.7410 - val_acc: 0.6810
Epoch 4/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.7369 -
acc: 0.6862 - val_loss: 1.2686 - val_acc: 0.4058
Epoch 5/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.5986 -
acc: 0.7520 - val_loss: 0.5492 - val_acc: 0.7652
Epoch 6/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.4807 -
acc: 0.8075 - val_loss: 0.5612 - val_acc: 0.7727
Epoch 7/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.4437 -
acc: 0.8478 - val_loss: 0.5127 - val_acc: 0.8324
Epoch 8/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.3448 -
acc: 0.8921 - val_loss: 0.4319 - val_acc: 0.8504
Epoch 9/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.3270 -
acc: 0.9007 - val_loss: 0.4478 - val_acc: 0.8646
Epoch 10/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.2576 -
acc: 0.9146 - val_loss: 0.2975 - val_acc: 0.9006
Epoch 11/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.2145 -
acc: 0.9268 - val_loss: 0.3361 - val_acc: 0.9002
Epoch 12/30
7352/7352 [==============================] - 138s 19ms/step - loss: 0.1917 -
acc: 0.9332 - val_loss: 0.3147 - val_acc: 0.9009
Epoch 13/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1934 -
acc: 0.9340 - val_loss: 0.3482 - val_acc: 0.8890
Epoch 14/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1755 -
acc: 0.9402 - val_loss: 0.4045 - val_acc: 0.8921
Epoch 15/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1868 -
acc: 0.9362 - val_loss: 0.3341 - val_acc: 0.9128
Epoch 16/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1909 -
acc: 0.9391 - val_loss: 0.3687 - val_acc: 0.9145
Epoch 17/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1932 -
acc: 0.9373 - val_loss: 0.3218 - val_acc: 0.9070
Epoch 18/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1643 -
acc: 0.9425 - val_loss: 0.3123 - val_acc: 0.9080
Epoch 19/30
7352/7352 [==============================] - 137s 19ms/step - loss: 0.1666 -
```

```
acc: 0.9438 - val_loss: 0.3939 - val_acc: 0.9182
Epoch 20/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1633 -
acc: 0.9423 - val_loss: 0.3127 - val_acc: 0.9155
Epoch 21/30
7352/7352 [==============================] - 136s 19ms/step - loss: 0.1758 -
acc: 0.9436 - val_loss: 0.4191 - val_acc: 0.8999
Epoch 22/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1579 -
acc: 0.9433 - val_loss: 0.4217 - val_acc: 0.8982
Epoch 23/30
7352/7352 [==============================] - 137s 19ms/step - loss: 0.1589 -
acc: 0.9441 - val_loss: 0.2486 - val_acc: 0.9006
Epoch 24/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1497 -
acc: 0.9445 - val_loss: 0.3153 - val_acc: 0.9023
Epoch 25/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.1422 -
acc: 0.9479 - val_loss: 0.3781 - val_acc: 0.9189
Epoch 26/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.1503 -
acc: 0.9499 - val_loss: 0.4217 - val_acc: 0.9030
Epoch 27/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1594 -
acc: 0.9431 - val_loss: 0.4035 - val_acc: 0.9104
Epoch 28/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1597 -
acc: 0.9410 - val_loss: 0.3851 - val_acc: 0.9087
Epoch 29/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1853 -
acc: 0.9434 - val_loss: 0.4077 - val_acc: 0.8853
Epoch 30/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1434 -
acc: 0.9461 - val_loss: 0.4164 - val_acc: 0.9046
```

Out[25]: <keras.callbacks.History at 0x135711a3da0>

In [26]:
```
# 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                  0      379       112        0                   0
STANDING                 0       70       461        1                   0
WALKING                  0        0         0      472                   0
WALKING_DOWNSTAIRS       0        0         0        5                 378
WALKING_UPSTAIRS         0        1         0        4                   0

Pred                WALKING_UPSTAIRS
True
LAYING                            0
SITTING                           0
STANDING                          0
WALKING                          24
WALKING_DOWNSTAIRS               37
WALKING_UPSTAIRS                466
```

In [27]:
```
score = model.evaluate(X_test, Y_test)
```

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

In [28]:
```
score
```

Out[28]:
```
[0.4163782881195881, 0.9046487953851374]
```

Model-5:

In [52]:
```

```

In [53]:
```
model = Sequential()
model.add(LSTM(100, input_shape=(timesteps, input_dim), kernel_initializer='gl
orot_normal' , return_sequences=True, bias_regularizer=reg))
model.add(BatchNormalization())
model.add(Dropout(0.70))
model.add(LSTM(50))
model.add(Dropout(0.70))
model.add(Dense(n_classes, activation='sigmoid'))
print("Model Summary: ")
model.summary()
```

```
Model Summary:
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_10 (LSTM)               (None, 128, 100)          44000
_________________________________________________________________
batch_normalization_5 (Batch (None, 128, 100)          400
_________________________________________________________________
dropout_9 (Dropout)          (None, 128, 100)          0
_________________________________________________________________
lstm_11 (LSTM)               (None, 50)                30200
_________________________________________________________________
dropout_10 (Dropout)         (None, 50)                0
_________________________________________________________________
dense_5 (Dense)              (None, 6)                 306
=================================================================
Total params: 74,906
Trainable params: 74,706
Non-trainable params: 200
_________________________________________________________________
```

In [56]:
```
# Compiling the model
model.compile(loss='binary_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
checkpoint_3 = ModelCheckpoint("model_7.h5",monitor="val_acc",mode="max",save_
best_only = True,verbose=1)
```

In [57]:
```
# Training the model
model.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=20,callbacks=[checkpoint_3])
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/20
7352/7352 [==============================] - 244s 33ms/step - loss: 1.7097 -
acc: 0.8483 - val_loss: 1.0533 - val_acc: 0.8947

Epoch 00001: val_acc improved from -inf to 0.89470, saving model to model_7.h
5
Epoch 2/20
7352/7352 [==============================] - 249s 34ms/step - loss: 0.6515 -
acc: 0.9053 - val_loss: 0.2413 - val_acc: 0.9536

Epoch 00002: val_acc improved from 0.89470 to 0.95357, saving model to model_
7.h5
Epoch 3/20
7352/7352 [==============================] - 240s 33ms/step - loss: 0.1730 -
acc: 0.9407 - val_loss: 0.1105 - val_acc: 0.9600

Epoch 00003: val_acc improved from 0.95357 to 0.96002, saving model to model_
7.h5
Epoch 4/20
7352/7352 [==============================] - 243s 33ms/step - loss: 0.1270 -
acc: 0.9586 - val_loss: 0.1410 - val_acc: 0.9484

Epoch 00004: val_acc did not improve from 0.96002
Epoch 5/20
7352/7352 [==============================] - 253s 34ms/step - loss: 0.1128 -
acc: 0.9617 - val_loss: 0.1013 - val_acc: 0.9673

Epoch 00005: val_acc improved from 0.96002 to 0.96725, saving model to model_
7.h5
Epoch 6/20
7352/7352 [==============================] - 247s 34ms/step - loss: 0.1016 -
acc: 0.9652 - val_loss: 0.0821 - val_acc: 0.9706

Epoch 00006: val_acc improved from 0.96725 to 0.97059, saving model to model_
7.h5
Epoch 7/20
7352/7352 [==============================] - 248s 34ms/step - loss: 0.0906 -
acc: 0.9697 - val_loss: 0.1346 - val_acc: 0.9522

Epoch 00007: val_acc did not improve from 0.97059
Epoch 8/20
7352/7352 [==============================] - 257s 35ms/step - loss: 0.0874 -
acc: 0.9704 - val_loss: 0.1169 - val_acc: 0.9579

Epoch 00008: val_acc did not improve from 0.97059
Epoch 9/20
7352/7352 [==============================] - 265s 36ms/step - loss: 0.0859 -
acc: 0.9700 - val_loss: 0.1319 - val_acc: 0.9574

Epoch 00009: val_acc did not improve from 0.97059
Epoch 10/20
7352/7352 [==============================] - 264s 36ms/step - loss: 0.0807 -
acc: 0.9726 - val_loss: 0.0992 - val_acc: 0.9663

Epoch 00010: val_acc did not improve from 0.97059
Epoch 11/20
```

```
7352/7352 [==============================] - 249s 34ms/step - loss: 0.0816 -
acc: 0.9730 - val_loss: 0.0948 - val_acc: 0.9680

Epoch 00011: val_acc did not improve from 0.97059
Epoch 12/20
7352/7352 [==============================] - 255s 35ms/step - loss: 0.0767 -
acc: 0.9729 - val_loss: 0.1639 - val_acc: 0.9501

Epoch 00012: val_acc did not improve from 0.97059
Epoch 13/20
7352/7352 [==============================] - 263s 36ms/step - loss: 0.0767 -
acc: 0.9724 - val_loss: 0.1193 - val_acc: 0.9649

Epoch 00013: val_acc did not improve from 0.97059
Epoch 14/20
7352/7352 [==============================] - 270s 37ms/step - loss: 0.0733 -
acc: 0.9738 - val_loss: 0.1060 - val_acc: 0.9657

Epoch 00014: val_acc did not improve from 0.97059
Epoch 15/20
7352/7352 [==============================] - 289s 39ms/step - loss: 0.0737 -
acc: 0.9738 - val_loss: 0.0743 - val_acc: 0.9751

Epoch 00015: val_acc improved from 0.97059 to 0.97512, saving model to model_
7.h5
Epoch 16/20
7352/7352 [==============================] - 300s 41ms/step - loss: 0.0722 -
acc: 0.9745 - val_loss: 0.1129 - val_acc: 0.9722

Epoch 00016: val_acc did not improve from 0.97512
Epoch 17/20
7352/7352 [==============================] - 295s 40ms/step - loss: 0.0723 -
acc: 0.9751 - val_loss: 0.0909 - val_acc: 0.9710

Epoch 00017: val_acc did not improve from 0.97512
Epoch 18/20
7352/7352 [==============================] - 358s 49ms/step - loss: 0.0701 -
acc: 0.9752 - val_loss: 0.1355 - val_acc: 0.9623

Epoch 00018: val_acc did not improve from 0.97512
Epoch 19/20
7352/7352 [==============================] - 329s 45ms/step - loss: 0.0715 -
acc: 0.9743 - val_loss: 0.0846 - val_acc: 0.9731

Epoch 00019: val_acc did not improve from 0.97512
Epoch 20/20
7352/7352 [==============================] - 275s 37ms/step - loss: 0.0684 -
acc: 0.9748 - val_loss: 0.0841 - val_acc: 0.9748

Epoch 00020: val_acc did not improve from 0.97512
```

Out[57]: <keras.callbacks.History at 0x15c9fe36f28>

In [58]:
```
model = load_model('model_7.h5')
```

In [59]:
```
# 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                 20      419        51        0                   0
STANDING                 0      116       416        0                   0
WALKING                  0        0         2      490                   3
WALKING_DOWNSTAIRS       0        0         0        0                 420
WALKING_UPSTAIRS         0        1         0       11                  10

Pred                WALKING_UPSTAIRS
True
LAYING                            0
SITTING                           1
STANDING                          0
WALKING                           1
WALKING_DOWNSTAIRS                0
WALKING_UPSTAIRS                449
```

In [60]:
```
score = model.evaluate(X_test, Y_test)
```

```
2947/2947 [==============================] - 18s 6ms/step
```

In [61]:
```
score
```

Out[61]:
```
[0.07431104327187013, 0.9751159407787497]
```

In [ ]:
```

```

Results(PrettyTable):

In [3]:
```
from prettytable import PrettyTable
x = PrettyTable()
x.field_names = ["Model","Description", "Test loss", "Test Accuracy"]
x.add_row(["1","1 Layer of LSTM(40)","0.5031", "0.8893"])
x.add_row(["2","2 layers of LSTM, BN, Categoricalcross_entropy","0.2544", "0.9
338"])
x.add_row(["3","1 Layer of LSTM(70)","0.4624", "0.8988"])
x.add_row(["4","1 Layer of LSTM(50)","0.4163", "0.9046"])
x.add_row(["5","2 layers of LSTM, BN, Binarycross_entropy","0.0743", "0.9751"
])
print(x)
```

```
+-------+-----------------------------------------------+-----------+-------
--------+
| Model |                  Description                  | Test loss | Test A
ccuracy |
+-------+-----------------------------------------------+-----------+-------
--------+
|   1   |              1 Layer of LSTM(40)              |   0.5031  |     0.
8893    |
|   2   | 2 layers of LSTM, BN, Categoricalcross_entropy |   0.2544  |     0.
9338    |
|   3   |              1 Layer of LSTM(70)              |   0.4624  |     0.
8988    |
|   4   |              1 Layer of LSTM(50)              |   0.4163  |     0.
9046    |
|   5   |   2 layers of LSTM, BN, Binarycross_entropy   |   0.0743  |     0.
9751    |
+-------+-----------------------------------------------+-----------+-------
--------+
```

Conclusions:

1. First i've used a single LSTM layer with 40 units, and got 0.88
2. Then i used 2 layers of LSTM follwed by Batch normalizationn with categorical cross entropy and got 0.93
3. For the 3rd and 4th model ,i used 1 Layer of LSTM with 70,50 units respectively and got a accuracy of 0.90
4. Now For 5th model i've used 2 Layers of LSTM with Binary cross entropy along with Batch normalization,with some dropout and got accuracy of 0.97

In [ ]:
```
```