In [3]:

```
import pandas as pd
import numpy as np
from prettytable import PrettyTable
```

In [4]:

```
# 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 [5]:

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

In [6]:

```
# 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 [7]:

```
# 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 [8]:

```
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 [9]:

```
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 [10]:

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

```
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\tensorboard\compat\tensorflow_stub\dtypes.py:541: FutureWarning: Passing (type, 1) or '1t
ype' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (t
ype, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\tensorboard\compat\tensorflow_stub\dtypes.py:542: FutureWarning: Passing (type, 1) or '1t
ype' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (t
ype, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\tensorboard\compat\tensorflow_stub\dtypes.py:543: FutureWarning: Passing (type, 1) or '1t
ype' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (t
ype, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\tensorboard\compat\tensorflow_stub\dtypes.py:544: FutureWarning: Passing (type, 1) or '1t
ype' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (t
ype, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\tensorboard\compat\tensorflow_stub\dtypes.py:545: FutureWarning: Passing (type, 1) or '1t
ype' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (t
ype, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\tensorboard\compat\tensorflow_stub\dtypes.py:550: FutureWarning: Passing (type, 1) or '1t
ype' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (t
ype, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

In [11]:

```
# Configuring a session
```

```
session_conf = tf.ConfigProto(
    intra_op_parallelism_threads=1,
    inter_op_parallelism_threads=1
)
```

In [12]:

```
# 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 [19]:

```
# Importing libraries
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers.core import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
```

In [12]:

```
# Initializing parameters

epochs = 35
batch_size = 32
n_hidden = 256
drop_out = 0.65
```

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\acer\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:12: F
utureWarning: Method .as_matrix will be removed in a future version. Use .values instead.
  if sys.path[0] == '':
c:\users\acer\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:11: F
utureWarning: Method .as_matrix will be removed in a future version. Use .values instead.
  # This is added back by InteractiveShellApp.init_path()
```

In [16]:

```
timesteps = len(X_train[0])
input_dim = len(X_train[0][0])
n_classes = _count_classes(Y_train)

print(timesteps)
print(input_dim)
print(len(X_train))
```

```
128
9
7352
```

In [24]:

```
X_train.shape
```

Out[24]:

Out[24]:

```
(7352, 128, 9)
```

- Defining the Architecture of LSTM

LSTM model with 1 layer

Example 1

In [17]:

```
import warnings
warnings.filterwarnings("ignore")

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

```
W1010 23:19:21.719681 11976 nn_ops.py:4224] Large dropout rate: 0.65 (>0.5). In TensorFlow 2.x,
dropout() uses dropout rate instead of keep_prob. Please ensure that this is intended.
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_2 (LSTM)                (None, 256)               272384
_________________________________________________________________
dropout_2 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 1542
=================================================================
Total params: 273,926
Trainable params: 273,926
Non-trainable params: 0
_________________________________________________________________
```

In [18]:

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

```
W1010 23:19:51.326107 11976 deprecation_wrapper.py:119] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-packages\keras\optimizers.py:790: Th
e name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

W1010 23:19:51.372969 11976 deprecation_wrapper.py:119] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\keras\backend\tensorflow_backend.py:3295: The name tf.log is deprecated. Please use tf.ma
th.log instead.
```

In [19]:

```
import warnings
warnings.filterwarnings("ignore")

from datetime import datetime
start = datetime.now()
```

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

print("Time taken : ", datetime.now() - start)
```

```
W1010 23:20:01.681306 11976 deprecation.py:323] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\tensorflow\python\ops\math_grad.py:1250: add_dispatch_support.<locals>.wrapper (from
tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/35
7352/7352 [==============================] - 142s 19ms/step - loss: 1.3411 - acc: 0.4002 - val_los
s: 1.2544 - val_acc: 0.4768
Epoch 2/35
7352/7352 [==============================] - 134s 18ms/step - loss: 1.2234 - acc: 0.4752 - val_los
s: 1.5577 - val_acc: 0.3244
Epoch 3/35
7352/7352 [==============================] - 137s 19ms/step - loss: 1.1260 - acc: 0.5276 - val_los
s: 1.1605 - val_acc: 0.5087
Epoch 4/35
7352/7352 [==============================] - 135s 18ms/step - loss: 1.1302 - acc: 0.5150 - val_los
s: 0.8060 - val_acc: 0.6057
Epoch 5/35
7352/7352 [==============================] - 135s 18ms/step - loss: 0.7858 - acc: 0.6295 - val_los
s: 0.7274 - val_acc: 0.6488
Epoch 6/35
7352/7352 [==============================] - 135s 18ms/step - loss: 0.6661 - acc: 0.7057 - val_los
s: 0.6510 - val_acc: 0.7418
Epoch 7/35
7352/7352 [==============================] - 135s 18ms/step - loss: 0.4786 - acc: 0.8263 - val_los
s: 0.4258 - val_acc: 0.8602
Epoch 8/35
7352/7352 [==============================] - 134s 18ms/step - loss: 0.3406 - acc: 0.8838 - val_los
s: 0.3102 - val_acc: 0.8931
Epoch 9/35
7352/7352 [==============================] - 134s 18ms/step - loss: 0.2619 - acc: 0.9127 - val_los
s: 0.3605 - val_acc: 0.8992
Epoch 10/35
7352/7352 [==============================] - 133s 18ms/step - loss: 0.2322 - acc: 0.9210 - val_los
s: 0.3511 - val_acc: 0.8867
Epoch 11/35
7352/7352 [==============================] - 133s 18ms/step - loss: 0.2069 - acc: 0.9280 - val_los
s: 0.6683 - val_acc: 0.8527
Epoch 12/35
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1941 - acc: 0.9354 - val_los
s: 0.3235 - val_acc: 0.9053
Epoch 13/35
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1800 - acc: 0.9363 - val_los
s: 0.2559 - val_acc: 0.9074
Epoch 14/35
7352/7352 [==============================] - 132s 18ms/step - loss: 0.1847 - acc: 0.9354 - val_los
s: 0.3230 - val_acc: 0.9080
Epoch 15/35
7352/7352 [==============================] - 133s 18ms/step - loss: 0.2477 - acc: 0.9240 - val_los
s: 0.2513 - val_acc: 0.9148
Epoch 16/35
7352/7352 [==============================] - 136s 18ms/step - loss: 0.1635 - acc: 0.9402 - val_los
s: 0.3573 - val_acc: 0.9074
Epoch 17/35
7352/7352 [==============================] - 132s 18ms/step - loss: 0.1527 - acc: 0.9479 - val_los
s: 0.3652 - val_acc: 0.9016
Epoch 18/35
7352/7352 [==============================] - 130s 18ms/step - loss: 0.1545 - acc: 0.9446 - val_los
s: 0.3818 - val_acc: 0.9053
Epoch 19/35
7352/7352 [==============================] - 132s 18ms/step - loss: 0.1592 - acc: 0.9431 - val_los
s: 0.3257 - val_acc: 0.8951
Epoch 20/35
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1404 - acc: 0.9456 - val_los
```

```
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1404 - acc: 0.9456 - val_los
s: 0.5671 - val_acc: 0.8734
Epoch 21/35
7352/7352 [==============================] - 132s 18ms/step - loss: 0.1400 - acc: 0.9463 - val_los
s: 0.3576 - val_acc: 0.8894
Epoch 22/35
7352/7352 [==============================] - 130s 18ms/step - loss: 0.1385 - acc: 0.9479 - val_los
s: 0.3086 - val_acc: 0.9257
Epoch 23/35
7352/7352 [==============================] - 131s 18ms/step - loss: 0.1698 - acc: 0.9457 - val_los
s: 0.2936 - val_acc: 0.9108
Epoch 24/35
7352/7352 [==============================] - 132s 18ms/step - loss: 0.1461 - acc: 0.9440 - val_los
s: 0.3550 - val_acc: 0.8989
Epoch 25/35
7352/7352 [==============================] - 132s 18ms/step - loss: 0.1493 - acc: 0.9456 - val_los
s: 0.3228 - val_acc: 0.9155
Epoch 26/35
7352/7352 [==============================] - 131s 18ms/step - loss: 0.1283 - acc: 0.9498 - val_los
s: 0.4807 - val_acc: 0.8728
Epoch 27/35
7352/7352 [==============================] - 131s 18ms/step - loss: 0.1207 - acc: 0.9512 - val_los
s: 0.5020 - val_acc: 0.9084
Epoch 28/35
7352/7352 [==============================] - 131s 18ms/step - loss: 0.1180 - acc: 0.9527 - val_los
s: 0.3718 - val_acc: 0.9046
Epoch 29/35
7352/7352 [==============================] - 132s 18ms/step - loss: 0.1385 - acc: 0.9495 - val_los
s: 0.3655 - val_acc: 0.9162
Epoch 30/35
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1372 - acc: 0.9504 - val_los
s: 0.3705 - val_acc: 0.9216
Epoch 31/35
7352/7352 [==============================] - 150s 20ms/step - loss: 0.1530 - acc: 0.9501 - val_los
s: 0.5002 - val_acc: 0.9006
Epoch 32/35
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1535 - acc: 0.9505 - val_los
s: 0.4029 - val_acc: 0.9009
Epoch 33/35
7352/7352 [==============================] - 144s 20ms/step - loss: 0.1345 - acc: 0.9508 - val_los
s: 0.3555 - val_acc: 0.9152
Epoch 34/35
7352/7352 [==============================] - 147s 20ms/step - loss: 0.1319 - acc: 0.9479 - val_los
s: 0.3275 - val_acc: 0.9199
Epoch 35/35
7352/7352 [==============================] - 141s 19ms/step - loss: 0.1498 - acc: 0.9495 - val_los
s: 0.6165 - val_acc: 0.9060
Time taken :  1:18:35.114140
```

In [20]:

```
# 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      375       115        1                   0
STANDING                 0       57       473        0                   0
WALKING                  0        2         2      477                   6
WALKING_DOWNSTAIRS       0        0         1       37                 376
WALKING_UPSTAIRS         0        0         0       39                   0

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            0
STANDING                           2
WALKING                            9
WALKING_DOWNSTAIRS                 6
WALKING_UPSTAIRS                 432
```

In [21]:

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

```
2947/2947 [==============================] - 9s 3ms/step
```

In [22]:

```
score
```

Out[22]:

```
[0.6165444772455196, 0.9060061079063454]
```

- 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

Example 2

In [23]:

```
# Initializing parameters

epochs = 30
batch_size = 32
n_hidden = 512
drop_out = 0.80
```

In [24]:

```
import warnings
warnings.filterwarnings("ignore")

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

```
W1011 00:51:38.137266 11976 nn_ops.py:4224] Large dropout rate: 0.8 (>0.5). In TensorFlow 2.x,
dropout() uses dropout rate instead of keep_prob. Please ensure that this is intended.
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_3 (LSTM)                (None, 512)               1069056
_________________________________________________________________
dropout_3 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_3 (Dense)              (None, 6)                 3078
=================================================================
Total params: 1,072,134
Trainable params: 1,072,134
Non-trainable params: 0
_________________________________________________________________
```

In [25]:

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

In [26]:

In [20]:

```
import warnings
warnings.filterwarnings("ignore")

from datetime import datetime
start = datetime.now()

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

print("Time taken : ", datetime.now() - start)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 629s 86ms/step - loss: 1.5089 - acc: 0.3402 - val_los
s: 1.3403 - val_acc: 0.4092
Epoch 2/30
7352/7352 [==============================] - 649s 88ms/step - loss: 1.5295 - acc: 0.3716 - val_los
s: 1.5504 - val_acc: 0.3739
Epoch 3/30
7352/7352 [==============================] - 629s 86ms/step - loss: 1.3803 - acc: 0.4115 - val_los
s: 1.3213 - val_acc: 0.4174
Epoch 4/30
7352/7352 [==============================] - 547s 74ms/step - loss: 1.1767 - acc: 0.4973 - val_los
s: 0.8094 - val_acc: 0.6030
Epoch 5/30
7352/7352 [==============================] - 289s 39ms/step - loss: 0.8387 - acc: 0.6269 - val_los
s: 0.7467 - val_acc: 0.6977
Epoch 6/30
7352/7352 [==============================] - 286s 39ms/step - loss: 0.7446 - acc: 0.6771 - val_los
s: 0.6599 - val_acc: 0.7119
Epoch 7/30
7352/7352 [==============================] - 296s 40ms/step - loss: 0.5344 - acc: 0.8040 - val_los
s: 0.5986 - val_acc: 0.7536
Epoch 8/30
7352/7352 [==============================] - 295s 40ms/step - loss: 0.3996 - acc: 0.8655 - val_los
s: 0.3836 - val_acc: 0.8704
Epoch 9/30
7352/7352 [==============================] - 297s 40ms/step - loss: 0.3145 - acc: 0.8942 - val_los
s: 0.4376 - val_acc: 0.8575
Epoch 10/30
7352/7352 [==============================] - 284s 39ms/step - loss: 0.2822 - acc: 0.9037 - val_los
s: 0.5159 - val_acc: 0.7900
Epoch 11/30
7352/7352 [==============================] - 288s 39ms/step - loss: 0.2558 - acc: 0.9158 - val_los
s: 0.3717 - val_acc: 0.8863
Epoch 12/30
7352/7352 [==============================] - 295s 40ms/step - loss: 0.2101 - acc: 0.9255 - val_los
s: 0.5566 - val_acc: 0.8890
Epoch 13/30
7352/7352 [==============================] - 354s 48ms/step - loss: 0.2035 - acc: 0.9305 - val_los
s: 0.5502 - val_acc: 0.8700
Epoch 14/30
7352/7352 [==============================] - 286s 39ms/step - loss: 0.1760 - acc: 0.9329 - val_los
s: 0.6543 - val_acc: 0.8738
Epoch 15/30
7352/7352 [==============================] - 283s 39ms/step - loss: 0.2171 - acc: 0.9350 - val_los
s: 0.4370 - val_acc: 0.9023
Epoch 16/30
7352/7352 [==============================] - 387s 53ms/step - loss: 0.2031 - acc: 0.9295 - val_los
s: 0.3562 - val_acc: 0.9023
Epoch 17/30
7352/7352 [==============================] - 562s 76ms/step - loss: 0.1515 - acc: 0.9418 - val_los
s: 1.5394 - val_acc: 0.8086
Epoch 18/30
7352/7352 [==============================] - 561s 76ms/step - loss: 0.1612 - acc: 0.9426 - val_los
s: 0.5296 - val_acc: 0.9033
Epoch 19/30
7352/7352 [==============================] - 597s 81ms/step - loss: 0.1517 - acc: 0.9430 - val_los
s: 0.3856 - val_acc: 0.9087
Epoch 20/30
7352/7352 [==============================] - 603s 82ms/step - loss: 0.1406 - acc: 0.9455 - val_los
```

```
s: 0.5601 - val_acc: 0.8985
Epoch 21/30
7352/7352 [==============================] - 600s 82ms/step - loss: 0.1743 - acc: 0.9423 - val_los
s: 0.3276 - val_acc: 0.9230
Epoch 22/30
7352/7352 [==============================] - 595s 81ms/step - loss: 0.1485 - acc: 0.9475 - val_los
s: 0.3797 - val_acc: 0.9070
Epoch 23/30
7352/7352 [==============================] - 594s 81ms/step - loss: nan - acc: 0.4944 - val_loss:
nan - val_acc: 0.1683
Epoch 24/30
7352/7352 [==============================] - 589s 80ms/step - loss: nan - acc: 0.1668 - val_loss:
nan - val_acc: 0.1683
Epoch 25/30
7352/7352 [==============================] - 588s 80ms/step - loss: nan - acc: 0.1668 - val_loss:
nan - val_acc: 0.1683
Epoch 26/30
7352/7352 [==============================] - 591s 80ms/step - loss: nan - acc: 0.1668 - val_loss:
nan - val_acc: 0.1683
Epoch 27/30
7352/7352 [==============================] - 596s 81ms/step - loss: nan - acc: 0.1668 - val_loss:
nan - val_acc: 0.1683
Epoch 28/30
7352/7352 [==============================] - 588s 80ms/step - loss: nan - acc: 0.1668 - val_loss:
nan - val_acc: 0.1683
Epoch 29/30
7352/7352 [==============================] - 590s 80ms/step - loss: nan - acc: 0.1668 - val_loss:
nan - val_acc: 0.1683
Epoch 30/30
7352/7352 [==============================] - 594s 81ms/step - loss: nan - acc: 0.1668 - val_loss:
nan - val_acc: 0.1683
Time taken :  3:59:08.041579
```

In [27]:

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

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

In [28]:

```
score = model.evaluate(X_test, Y_test,verbose=0)
```

```
2947/2947 [==============================] - 31s 11ms/step
```

In [32]:

```
print(score[0])
print(score[1])
```

```
nan
0.168306752629793
```

Example 3

In [33]:

```
# Initializing parameters

epochs = 30
batch_size = 32
n_hidden = 128
```

```
drop_out = 0.50
```

In [34]:

```
import warnings
warnings.filterwarnings("ignore")

# Initiliazing the sequential model
model_1 = Sequential()
# Configuring the parameters
model_1.add(LSTM(n_hidden, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model_1.add(Dropout(drop_out))
# Adding a dense output layer with sigmoid activation
model_1.add(Dense(n_classes, activation='sigmoid'))
model_1.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 [35]:

```
import warnings
warnings.filterwarnings("ignore")
# Compiling the model
model_1.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

In [36]:

```
import warnings
warnings.filterwarnings("ignore")

from datetime import datetime
start = datetime.now()

# Training the model
model_1.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=epochs)

print("Time taken : ", datetime.now() - start)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 29s 4ms/step - loss: 1.3071 - acc: 0.4188 - val_loss:
1.2370 - val_acc: 0.4123
Epoch 2/30
7352/7352 [==============================] - 28s 4ms/step - loss: 1.1652 - acc: 0.4850 - val_loss:
1.1890 - val_acc: 0.4890
Epoch 3/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.9696 - acc: 0.5760 - val_loss:
0.8628 - val_acc: 0.6468
Epoch 4/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.7924 - acc: 0.6372 - val_loss:
0.8567 - val_acc: 0.5969
Epoch 5/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.6695 - acc: 0.6858 - val_loss:
0.6848 - val_acc: 0.6780
```

```
Epoch 6/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.5966 - acc: 0.7522 - val_loss:
0.5937 - val_acc: 0.8042
Epoch 7/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.3786 - acc: 0.8674 - val_loss:
0.5302 - val_acc: 0.8198
Epoch 8/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2979 - acc: 0.8979 - val_loss:
0.3259 - val_acc: 0.8931
Epoch 9/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.3337 - acc: 0.8795 - val_loss:
0.3682 - val_acc: 0.8901
Epoch 10/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2501 - acc: 0.9129 - val_loss:
0.2773 - val_acc: 0.8982
Epoch 11/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2554 - acc: 0.9067 - val_loss:
0.6779 - val_acc: 0.7635
Epoch 12/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.2033 - acc: 0.9240 - val_loss:
0.3708 - val_acc: 0.8490
Epoch 13/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1900 - acc: 0.9323 - val_loss:
0.2243 - val_acc: 0.9148
Epoch 14/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1758 - acc: 0.9358 - val_loss:
0.2729 - val_acc: 0.8941
Epoch 15/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1898 - acc: 0.9308 - val_loss:
0.3019 - val_acc: 0.9203
Epoch 16/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1707 - acc: 0.9400 - val_loss:
0.2875 - val_acc: 0.9253
Epoch 17/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1665 - acc: 0.9369 - val_loss:
0.2664 - val_acc: 0.9097
Epoch 18/30
7352/7352 [==============================] - 31s 4ms/step - loss: 0.1681 - acc: 0.9363 - val_loss:
0.2658 - val_acc: 0.9101
Epoch 19/30
7352/7352 [==============================] - 34s 5ms/step - loss: 0.1488 - acc: 0.9433 - val_loss:
0.3019 - val_acc: 0.9121
Epoch 20/30
7352/7352 [==============================] - 31s 4ms/step - loss: 0.1459 - acc: 0.9433 - val_loss:
0.2904 - val_acc: 0.9152
Epoch 21/30
7352/7352 [==============================] - 33s 4ms/step - loss: 0.1422 - acc: 0.9475 - val_loss:
0.2684 - val_acc: 0.9145
Epoch 22/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1470 - acc: 0.9455 - val_loss:
0.2619 - val_acc: 0.9196
Epoch 23/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1727 - acc: 0.9442 - val_loss:
0.2196 - val_acc: 0.9148
Epoch 24/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1361 - acc: 0.9483 - val_loss:
0.2314 - val_acc: 0.9182
Epoch 25/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1315 - acc: 0.9464 - val_loss:
0.2499 - val_acc: 0.9199
Epoch 26/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1386 - acc: 0.9445 - val_loss:
0.2594 - val_acc: 0.9213
Epoch 27/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1299 - acc: 0.9493 - val_loss:
0.3200 - val_acc: 0.9162
Epoch 28/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1312 - acc: 0.9480 - val_loss:
0.3058 - val_acc: 0.9203
Epoch 29/30
7352/7352 [==============================] - 29s 4ms/step - loss: 0.1295 - acc: 0.9457 - val_loss:
0.3183 - val_acc: 0.9145
Epoch 30/30
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1247 - acc: 0.9502 - val_loss:
0.3031 - val_acc: 0.9260
Time taken :  0:14:24.212183
```

In [37]:

```
score_1 = model_1.evaluate(X_test, Y_test,verbose=0)
```

In [39]:

```
print("Test Score " , score_1[0])
print("Test Accuracy " , score_1[1])
```

```
Test Score  0.3031002142316866
Test Accuracy  0.9260264675941635
```

Example 4

In [15]:

```
# Initializing parameters

epochs = 25
batch_size = 128
n_hidden = 512
drop_out = 0.80
```

In [17]:

```
import warnings
warnings.filterwarnings("ignore")

# Initiliazing the sequential model
model_1 = Sequential()
# Configuring the parameters
model_1.add(LSTM(n_hidden, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model_1.add(Dropout(drop_out))
# Adding a dense output layer with sigmoid activation
model_1.add(Dense(n_classes, activation='sigmoid'))
model_1.summary()
```

```
W1019 20:37:39.842547 13684 nn_ops.py:4224] Large dropout rate: 0.8 (>0.5). In TensorFlow 2.x,
dropout() uses dropout rate instead of keep_prob. Please ensure that this is intended.
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_2 (LSTM)                (None, 512)               1069056
_________________________________________________________________
dropout_2 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 3078
=================================================================
Total params: 1,072,134
Trainable params: 1,072,134
Non-trainable params: 0
_________________________________________________________________
```

In [18]:

```
import warnings
warnings.filterwarnings("ignore")
# Compiling the model
model_1.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

```
W1019 20:37:46.794115 13684 deprecation_wrapper.py:119] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-packages\keras\optimizers.py:790: Th
e name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.
```

```
W1019 20:37:46.840937 13684 deprecation_wrapper.py:119] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\keras\backend\tensorflow_backend.py:3295: The name tf.log is deprecated. Please use tf.ma
th.log instead.
```

In [19]:

```
import warnings
warnings.filterwarnings("ignore")

from datetime import datetime
start = datetime.now()

# Training the model
model_1.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=epochs)

print("Time taken : ", datetime.now() - start)
```

```
W1019 20:37:58.244091 13684 deprecation.py:323] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\tensorflow\python\ops\math_grad.py:1250: add_dispatch_support.<locals>.wrapper (from
tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where

Train on 7352 samples, validate on 2947 samples
Epoch 1/25
7352/7352 [==============================] - 355s 48ms/step - loss: 1.4378 - acc: 0.3539 - val_los
s: 1.5373 - val_acc: 0.3488
Epoch 2/25
7352/7352 [==============================] - 361s 49ms/step - loss: 1.3609 - acc: 0.4019 - val_los
s: 1.5678 - val_acc: 0.3420
Epoch 3/25
7352/7352 [==============================] - 362s 49ms/step - loss: 1.6731 - acc: 0.2920 - val_los
s: 1.4541 - val_acc: 0.3414
Epoch 4/25
7352/7352 [==============================] - 357s 49ms/step - loss: 1.4820 - acc: 0.3636 - val_los
s: 1.7327 - val_acc: 0.2050
Epoch 5/25
7352/7352 [==============================] - 361s 49ms/step - loss: 1.4496 - acc: 0.4004 - val_los
s: 1.4869 - val_acc: 0.3580
Epoch 6/25
7352/7352 [==============================] - 405s 55ms/step - loss: 1.2548 - acc: 0.4641 - val_los
s: 1.2893 - val_acc: 0.4917
Epoch 7/25
7352/7352 [==============================] - 361s 49ms/step - loss: 1.4773 - acc: 0.3862 - val_los
s: 1.7715 - val_acc: 0.1890
Epoch 8/25
7352/7352 [==============================] - 369s 50ms/step - loss: 1.2936 - acc: 0.4557 - val_los
s: 1.4750 - val_acc: 0.3845
Epoch 9/25
7352/7352 [==============================] - 352s 48ms/step - loss: 1.2332 - acc: 0.4815 - val_los
s: 1.4504 - val_acc: 0.3655
Epoch 10/25
7352/7352 [==============================] - 343s 47ms/step - loss: 1.0042 - acc: 0.5492 - val_los
s: 1.0693 - val_acc: 0.5117
Epoch 11/25
7352/7352 [==============================] - 168s 23ms/step - loss: 0.8862 - acc: 0.5947 - val_los
s: 0.8713 - val_acc: 0.5955
Epoch 12/25
7352/7352 [==============================] - 168s 23ms/step - loss: 0.8103 - acc: 0.6318 - val_los
s: 0.7218 - val_acc: 0.6980
Epoch 13/25
7352/7352 [==============================] - 170s 23ms/step - loss: 0.7430 - acc: 0.6801 - val_los
s: 0.7840 - val_acc: 0.6413
Epoch 14/25
7352/7352 [==============================] - 166s 23ms/step - loss: 0.6529 - acc: 0.7398 - val_los
s: 0.5445 - val_acc: 0.7998
Epoch 15/25
7352/7352 [==============================] - 167s 23ms/step - loss: 0.5400 - acc: 0.7982 - val_los
```

```
7352/7352 [==============================] - 167s 23ms/step - loss: 0.5400 - acc: 0.7982 - val_los
s: 0.6705 - val_acc: 0.7516
Epoch 16/25
7352/7352 [==============================] - 167s 23ms/step - loss: 0.5816 - acc: 0.8138 - val_los
s: 0.4972 - val_acc: 0.8375
Epoch 17/25
7352/7352 [==============================] - 169s 23ms/step - loss: 0.4046 - acc: 0.8672 - val_los
s: 0.4473 - val_acc: 0.8198
Epoch 18/25
7352/7352 [==============================] - 171s 23ms/step - loss: 0.3540 - acc: 0.8828 - val_los
s: 0.3230 - val_acc: 0.8741
Epoch 19/25
7352/7352 [==============================] - 172s 23ms/step - loss: 0.2690 - acc: 0.9070 - val_los
s: 0.3315 - val_acc: 0.8639
Epoch 20/25
7352/7352 [==============================] - 172s 23ms/step - loss: 0.2292 - acc: 0.9138 - val_los
s: 0.3364 - val_acc: 0.8890
Epoch 21/25
7352/7352 [==============================] - 169s 23ms/step - loss: 0.3755 - acc: 0.8957 - val_los
s: 0.5657 - val_acc: 0.7964
Epoch 22/25
7352/7352 [==============================] - 172s 23ms/step - loss: 0.2502 - acc: 0.9119 - val_los
s: 0.5957 - val_acc: 0.7771
Epoch 23/25
7352/7352 [==============================] - 170s 23ms/step - loss: 0.3712 - acc: 0.8890 - val_los
s: 1.3730 - val_acc: 0.4079
Epoch 24/25
7352/7352 [==============================] - 178s 24ms/step - loss: 1.4716 - acc: 0.3187 - val_los
s: 1.4622 - val_acc: 0.3081
Epoch 25/25
7352/7352 [==============================] - 173s 23ms/step - loss: 1.0056 - acc: 0.6070 - val_los
s: 0.3739 - val_acc: 0.8673
Time taken :  1:43:00.639062
```

In [20]:

```
score_1 = model_1.evaluate(X_test, Y_test,verbose=0)
```

In [21]:

```
print("Test Score " , score_1[0])
print("Test Accuracy " , score_1[1])
```

```
Test Score  0.37387920911103095
Test Accuracy  0.8673227010519172
```

LSTM model with 2 layers

Example 1

In [43]:

```
epochs = 30
batch_size= 32
n_hidden_layer1 = 128
n_hidden_layer2 =64
drop_out_1 = 0.2
drop_out_2 = 0.5
```

In [44]:

```
from keras.layers.normalization import BatchNormalization

# Initiliazing the sequential model
model_2 = Sequential()
# Configuring the parameters
model_2.add(LSTM(n_hidden_layer1, return_sequences=True, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model_2.add(Dropout(drop_out_1))
# Adding batch normalization
```

```
model_2.add(BatchNormalization())

model_2.add(LSTM(n_hidden_layer2))
# Adding a dropout layer
model_2.add(Dropout(drop_out_2))
# Adding a dense output layer with sigmoid activation
model_2.add(Dense(n_classes, activation='sigmoid'))
model_2.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_5 (LSTM)                (None, 128, 128)          70656
_________________________________________________________________
dropout_5 (Dropout)          (None, 128, 128)          0
_________________________________________________________________
batch_normalization_1 (Batch (None, 128, 128)          512
_________________________________________________________________
lstm_6 (LSTM)                (None, 64)                49408
_________________________________________________________________
dropout_6 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_5 (Dense)              (None, 6)                 390
=================================================================
Total params: 120,966
Trainable params: 120,710
Non-trainable params: 256
_________________________________________________________________
```

In [45]:

```
# Compiling the model
model_2.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

In [46]:

```
import warnings
warnings.filterwarnings("ignore")

from datetime import datetime
start = datetime.now()

# Training the model
model_2.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=epochs)

print("Time taken : ", datetime.now() - start)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 110s 15ms/step - loss: 0.7840 - acc: 0.7444 - val_los
s: 0.4422 - val_acc: 0.8663
Epoch 2/30
7352/7352 [==============================] - 133s 18ms/step - loss: 0.3214 - acc: 0.9034 - val_los
s: 0.4218 - val_acc: 0.8537
Epoch 3/30
7352/7352 [==============================] - 132s 18ms/step - loss: 0.2339 - acc: 0.9226 - val_los
s: 0.2183 - val_acc: 0.9203
Epoch 4/30
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1915 - acc: 0.9319 - val_los
s: 0.2653 - val_acc: 0.9053
Epoch 5/30
7352/7352 [==============================] - 136s 19ms/step - loss: 0.1769 - acc: 0.9385 - val_los
s: 0.2450 - val_acc: 0.9067
Epoch 6/30
7352/7352 [==============================] - 140s 19ms/step - loss: 0.1631 - acc: 0.9380 - val_los
s: 0.2529 - val_acc: 0.8982
Epoch 7/30
7352/7352 [==============================] - 145s 20ms/step - loss: 0.1516 - acc: 0.9423 - val_los
```

```
s: 0.2489 - val_acc: 0.9189
Epoch 8/30
7352/7352 [==============================] - 136s 18ms/step - loss: 0.1495 - acc: 0.9434 - val_los
s: 0.2659 - val_acc: 0.9084
Epoch 9/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1442 - acc: 0.9431 - val_los
s: 0.3140 - val_acc: 0.9030
Epoch 10/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1442 - acc: 0.9440 - val_los
s: 0.1962 - val_acc: 0.9311
Epoch 11/30
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1331 - acc: 0.9437 - val_los
s: 0.3383 - val_acc: 0.8982
Epoch 12/30
7352/7352 [==============================] - 132s 18ms/step - loss: 0.1342 - acc: 0.9460 - val_los
s: 0.2190 - val_acc: 0.9237
Epoch 13/30
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1272 - acc: 0.9480 - val_los
s: 0.2696 - val_acc: 0.9094
Epoch 14/30
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1330 - acc: 0.9475 - val_los
s: 0.2646 - val_acc: 0.9162
Epoch 15/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1294 - acc: 0.9493 - val_los
s: 0.3072 - val_acc: 0.9226
Epoch 16/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1289 - acc: 0.9490 - val_los
s: 0.2725 - val_acc: 0.9141
Epoch 17/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1235 - acc: 0.9512 - val_los
s: 0.3073 - val_acc: 0.9335
Epoch 18/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1259 - acc: 0.9499 - val_los
s: 0.2598 - val_acc: 0.9247
Epoch 19/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1203 - acc: 0.9506 - val_los
s: 0.2656 - val_acc: 0.9203
Epoch 20/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1313 - acc: 0.9498 - val_los
s: 0.2134 - val_acc: 0.9301
Epoch 21/30
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1172 - acc: 0.9532 - val_los
s: 0.3513 - val_acc: 0.8867
Epoch 22/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1375 - acc: 0.9486 - val_los
s: 0.3127 - val_acc: 0.9158
Epoch 23/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1176 - acc: 0.9531 - val_los
s: 0.2179 - val_acc: 0.9199
Epoch 24/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1229 - acc: 0.9508 - val_los
s: 0.2534 - val_acc: 0.9267
Epoch 25/30
7352/7352 [==============================] - 135s 18ms/step - loss: 0.1163 - acc: 0.9521 - val_los
s: 0.2754 - val_acc: 0.9209
Epoch 26/30
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1193 - acc: 0.9513 - val_los
s: 0.2492 - val_acc: 0.9189
Epoch 27/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1109 - acc: 0.9550 - val_los
s: 0.2193 - val_acc: 0.9372
Epoch 28/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1190 - acc: 0.9525 - val_los
s: 0.3456 - val_acc: 0.9253
Epoch 29/30
7352/7352 [==============================] - 133s 18ms/step - loss: 0.1084 - acc: 0.9548 - val_los
s: 0.3434 - val_acc: 0.9118
Epoch 30/30
7352/7352 [==============================] - 134s 18ms/step - loss: 0.1205 - acc: 0.9538 - val_los
s: 0.2833 - val_acc: 0.9243
Time taken :  1:06:56.079307
```

In [47]:

```
score_2 = model_2.evaluate(X_test, Y_test,verbose=0)
```

In [48]:

```
print("Test Score " , score_2[0])
print("Test Accuracy " , score_2[1])
```

```
Test Score  0.28329760189572595
Test Accuracy  0.9243298269426535
```

Example 2

In [49]:

```
epochs = 50
batch_size= 64
n_hidden_layer1 = 32
n_hidden_layer2 =64
drop_out_1 = 0.5
drop_out_2 = 0.5
```

In [50]:

```
from keras.layers.normalization import BatchNormalization

# Initiliazing the sequential model
model_3 = Sequential()
# Configuring the parameters
model_3.add(LSTM(n_hidden_layer1, return_sequences=True, input_shape=(timesteps, input_dim)))
# Adding a dropout layer
model_3.add(Dropout(drop_out_1))
# Adding batch normalization
model_3.add(BatchNormalization())

model_3.add(LSTM(n_hidden_layer2))
# Adding a dropout layer
model_3.add(Dropout(drop_out_2))
# Adding a dense output layer with sigmoid activation
model_3.add(Dense(n_classes, activation='sigmoid'))
model_3.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_7 (LSTM)                (None, 128, 32)           5376
_________________________________________________________________
dropout_7 (Dropout)          (None, 128, 32)           0
_________________________________________________________________
batch_normalization_2 (Batch (None, 128, 32)           128
_________________________________________________________________
lstm_8 (LSTM)                (None, 64)                24832
_________________________________________________________________
dropout_8 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_6 (Dense)              (None, 6)                 390
=================================================================
Total params: 30,726
Trainable params: 30,662
Non-trainable params: 64
_________________________________________________________________
```

In [51]:

```
# Compiling the model
model_3.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

In [52]:

```
import warnings
```

```
warnings.filterwarnings("ignore")

from datetime import datetime
start = datetime.now()

# Training the model
model_3.fit(X_train,
          Y_train,
          batch_size=batch_size,
          validation_data=(X_test, Y_test),
          epochs=epochs)

print("Time taken : ", datetime.now() - start)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/50
7352/7352 [==============================] - 54s 7ms/step - loss: 1.1300 - acc: 0.5647 - val_loss:
0.9103 - val_acc: 0.6525
Epoch 2/50
7352/7352 [==============================] - 58s 8ms/step - loss: 0.7584 - acc: 0.6930 - val_loss:
0.7881 - val_acc: 0.6451
Epoch 3/50
7352/7352 [==============================] - 52s 7ms/step - loss: 0.5752 - acc: 0.7553 - val_loss:
0.6252 - val_acc: 0.7723
Epoch 4/50
7352/7352 [==============================] - 50s 7ms/step - loss: 0.4374 - acc: 0.8455 - val_loss:
0.4508 - val_acc: 0.8473
Epoch 5/50
7352/7352 [==============================] - 50s 7ms/step - loss: 0.3366 - acc: 0.8964 - val_loss:
0.4305 - val_acc: 0.8490
Epoch 6/50
7352/7352 [==============================] - 50s 7ms/step - loss: 0.2598 - acc: 0.9193 - val_loss:
0.3325 - val_acc: 0.8748
Epoch 7/50
7352/7352 [==============================] - 50s 7ms/step - loss: 0.2251 - acc: 0.9214 - val_loss:
0.3800 - val_acc: 0.8809
Epoch 8/50
7352/7352 [==============================] - 49s 7ms/step - loss: 0.2043 - acc: 0.9272 - val_loss:
0.3478 - val_acc: 0.8884
Epoch 9/50
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1899 - acc: 0.9314 - val_loss:
0.3101 - val_acc: 0.8996
Epoch 10/50
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1803 - acc: 0.9357 - val_loss:
0.2658 - val_acc: 0.9063
Epoch 11/50
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1639 - acc: 0.9378 - val_loss:
0.2617 - val_acc: 0.9111
Epoch 12/50
7352/7352 [==============================] - 49s 7ms/step - loss: 0.1631 - acc: 0.9434 - val_loss:
0.3102 - val_acc: 0.9162
Epoch 13/50
7352/7352 [==============================] - 49s 7ms/step - loss: 0.1570 - acc: 0.9418 - val_loss:
0.3238 - val_acc: 0.8992
Epoch 14/50
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1431 - acc: 0.9476 - val_loss:
0.2106 - val_acc: 0.9267
Epoch 15/50
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1448 - acc: 0.9468 - val_loss:
0.3298 - val_acc: 0.9057
Epoch 16/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1367 - acc: 0.9479 - val_loss:
0.3993 - val_acc: 0.8982
Epoch 17/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1368 - acc: 0.9476 - val_loss:
0.3545 - val_acc: 0.9111
Epoch 18/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1318 - acc: 0.9493 - val_loss:
0.2902 - val_acc: 0.9162
Epoch 19/50
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1266 - acc: 0.9486 - val_loss:
0.3716 - val_acc: 0.9030
Epoch 20/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1290 - acc: 0.9506 - val_loss:
0.3003 - val_acc: 0.9155
Epoch 21/50
```

```
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1330 - acc: 0.9476 - val_loss:
0.3965 - val_acc: 0.9002
Epoch 22/50
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1276 - acc: 0.9491 - val_loss:
0.3151 - val_acc: 0.9002
Epoch 23/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1319 - acc: 0.9474 - val_loss:
0.3492 - val_acc: 0.9189
Epoch 24/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1191 - acc: 0.9520 - val_loss:
0.3965 - val_acc: 0.9050
Epoch 25/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1247 - acc: 0.9524 - val_loss:
0.3338 - val_acc: 0.9250
Epoch 26/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1182 - acc: 0.9517 - val_loss:
0.3678 - val_acc: 0.9189
Epoch 27/50
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1192 - acc: 0.9501 - val_loss:
0.3613 - val_acc: 0.9101
Epoch 28/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1165 - acc: 0.9543 - val_loss:
0.2805 - val_acc: 0.9179
Epoch 29/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1170 - acc: 0.9531 - val_loss:
0.3283 - val_acc: 0.9277
Epoch 30/50
7352/7352 [==============================] - 50s 7ms/step - loss: 0.1201 - acc: 0.9513 - val_loss:
0.2991 - val_acc: 0.9233
Epoch 31/50
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1139 - acc: 0.9521 - val_loss:
0.3705 - val_acc: 0.9148
Epoch 32/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1133 - acc: 0.9525 - val_loss:
0.4109 - val_acc: 0.9070
Epoch 33/50
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1149 - acc: 0.9548 - val_loss:
0.3748 - val_acc: 0.9138
Epoch 34/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1159 - acc: 0.9516 - val_loss:
0.4341 - val_acc: 0.9141
Epoch 35/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1096 - acc: 0.9555 - val_loss:
0.4302 - val_acc: 0.9104
Epoch 36/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1141 - acc: 0.9532 - val_loss:
0.3597 - val_acc: 0.9270
Epoch 37/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1155 - acc: 0.9532 - val_loss:
0.4360 - val_acc: 0.9036
Epoch 38/50
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1089 - acc: 0.9546 - val_loss:
0.3867 - val_acc: 0.9057
Epoch 39/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1057 - acc: 0.9551 - val_loss:
0.4418 - val_acc: 0.9226
Epoch 40/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1044 - acc: 0.9547 - val_loss:
0.4893 - val_acc: 0.9108
Epoch 41/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1030 - acc: 0.9546 - val_loss:
0.4605 - val_acc: 0.9233
Epoch 42/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1039 - acc: 0.9576 - val_loss:
0.4518 - val_acc: 0.9158
Epoch 43/50
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1023 - acc: 0.9554 - val_loss:
0.4676 - val_acc: 0.9175
Epoch 44/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1059 - acc: 0.9548 - val_loss:
0.5450 - val_acc: 0.9074
Epoch 45/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.0935 - acc: 0.9585 - val_loss:
0.4906 - val_acc: 0.9121
Epoch 46/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.0996 - acc: 0.9558 - val_loss:
0.5014 - val_acc: 0.9101
```

```
Epoch 47/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.0985 - acc: 0.9592 - val_loss:
0.5265 - val_acc: 0.9125
Epoch 48/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.0981 - acc: 0.9612 - val_loss:
0.4949 - val_acc: 0.9267
Epoch 49/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.0963 - acc: 0.9614 - val_loss:
0.4187 - val_acc: 0.9237
Epoch 50/50
7352/7352 [==============================] - 51s 7ms/step - loss: 0.0951 - acc: 0.9601 - val_loss:
0.4538 - val_acc: 0.9152
Time taken :  0:42:33.388274
```

Using CNN

Model 1

In [22]:

```
# Importing libraries
import pandas as pd
from matplotlib import pyplot
from sklearn.preprocessing import StandardScaler
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers import Dropout
from keras.layers.convolutional import Conv1D
from keras.layers.convolutional import MaxPooling1D
from keras.utils import to_categorical
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers.core import Dense, Dropout
```

In [25]:

```
epochs = 25
batch_size= 128
```

In [26]:

```
model = Sequential()
model.add(Conv1D(filters=32, kernel_size=3, activation='relu',kernel_initializer='he_uniform',input
_shape=(128,9)))
model.add(Conv1D(filters=32, kernel_size=3, activation='relu',kernel_initializer='he_uniform'))
model.add(Dropout(0.6))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(50, activation='relu'))
model.add(Dense(6, activation='softmax'))
model.summary()
```

```
W1020 11:39:41.144582  4124 deprecation_wrapper.py:119] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\keras\backend\tensorflow_backend.py:4138: The name tf.random_uniform is deprecated. Pleas
e use tf.random.uniform instead.

W1020 11:39:41.472668  4124 deprecation_wrapper.py:119] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\keras\backend\tensorflow_backend.py:133: The name tf.placeholder_with_default is
deprecated. Please use tf.compat.v1.placeholder_with_default instead.

W1020 11:39:41.472668  4124 deprecation.py:506] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\keras\backend\tensorflow_backend.py:3445: 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 - keep_prob`.
W1020 11:39:41.472668  4124 nn_ops.py:4224] Large dropout rate: 0.6 (>0.5). In TensorFlow 2.x,
```

```
dropout() uses dropout rate instead of keep_prob. Please ensure that this is intended.
W1020 11:39:41.519531  4124 deprecation_wrapper.py:119] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\keras\backend\tensorflow_backend.py:3976: The name tf.nn.max_pool is deprecated. Please u
se tf.nn.max_pool2d instead.
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_1 (Conv1D)            (None, 126, 32)           896
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 124, 32)           3104
_________________________________________________________________
dropout_1 (Dropout)          (None, 124, 32)           0
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 62, 32)            0
_________________________________________________________________
flatten_1 (Flatten)          (None, 1984)              0
_________________________________________________________________
dense_1 (Dense)              (None, 50)                99250
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 306
=================================================================
Total params: 103,556
Trainable params: 103,556
Non-trainable params: 0
_________________________________________________________________
```

In [27]:

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

```
W1020 11:39:48.689685  4124 deprecation_wrapper.py:119] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-packages\keras\optimizers.py:790: Th
e name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

W1020 11:39:48.736552  4124 deprecation_wrapper.py:119] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\keras\backend\tensorflow_backend.py:3295: The name tf.log is deprecated. Please use tf.ma
th.log instead.
```

In [28]:

```
import warnings
warnings.filterwarnings("ignore")

from datetime import datetime
start = datetime.now()

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

print("Time taken : ", datetime.now() - start)
```

```
W1020 11:39:52.860619  4124 deprecation.py:323] From
c:\users\acer\appdata\local\programs\python\python37\lib\site-
packages\tensorflow\python\ops\math_grad.py:1250: add_dispatch_support.<locals>.wrapper (from
tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/25
7352/7352 [==============================] - 8s 1ms/step - loss: 0.7660 - acc: 0.6827 - val_loss:
0.6110 - val_acc: 0.7811
Epoch 2/25
```

```
7352/7352 [==============================] - 6s 880us/step - loss: 0.3490 - acc: 0.8629 -
val_loss: 0.4754 - val_acc: 0.8480
Epoch 3/25
7352/7352 [==============================] - 6s 875us/step - loss: 0.2435 - acc: 0.9063 -
val_loss: 0.3927 - val_acc: 0.8670
Epoch 4/25
7352/7352 [==============================] - 6s 875us/step - loss: 0.1837 - acc: 0.9310 -
val_loss: 0.3400 - val_acc: 0.8924
Epoch 5/25
7352/7352 [==============================] - 6s 884us/step - loss: 0.1522 - acc: 0.9406 -
val_loss: 0.3324 - val_acc: 0.8870
Epoch 6/25
7352/7352 [==============================] - 7s 905us/step - loss: 0.1385 - acc: 0.9423 -
val_loss: 0.3143 - val_acc: 0.8958
Epoch 7/25
7352/7352 [==============================] - 6s 880us/step - loss: 0.1210 - acc: 0.9501 -
val_loss: 0.2878 - val_acc: 0.9111
Epoch 8/25
7352/7352 [==============================] - 6s 875us/step - loss: 0.1155 - acc: 0.9506 -
val_loss: 0.2957 - val_acc: 0.9009
Epoch 9/25
7352/7352 [==============================] - 6s 873us/step - loss: 0.1123 - acc: 0.9533 -
val_loss: 0.3122 - val_acc: 0.9026
Epoch 10/25
7352/7352 [==============================] - 6s 871us/step - loss: 0.1045 - acc: 0.9538 -
val_loss: 0.3223 - val_acc: 0.9036
Epoch 11/25
7352/7352 [==============================] - 7s 908us/step - loss: 0.0998 - acc: 0.9558 -
val_loss: 0.3564 - val_acc: 0.8982
Epoch 12/25
7352/7352 [==============================] - 7s 900us/step - loss: 0.0955 - acc: 0.9596 -
val_loss: 0.3132 - val_acc: 0.9074
Epoch 13/25
7352/7352 [==============================] - 6s 873us/step - loss: 0.0933 - acc: 0.9584 -
val_loss: 0.3201 - val_acc: 0.9077
Epoch 14/25
7352/7352 [==============================] - 7s 902us/step - loss: 0.1010 - acc: 0.9561 -
val_loss: 0.3365 - val_acc: 0.9009
Epoch 15/25
7352/7352 [==============================] - 7s 903us/step - loss: 0.0894 - acc: 0.9599 -
val_loss: 0.3479 - val_acc: 0.9040
Epoch 16/25
7352/7352 [==============================] - 7s 916us/step - loss: 0.0799 - acc: 0.9653 -
val_loss: 0.3701 - val_acc: 0.9074
Epoch 17/25
7352/7352 [==============================] - 7s 903us/step - loss: 0.0788 - acc: 0.9663 -
val_loss: 0.3970 - val_acc: 0.9070
Epoch 18/25
7352/7352 [==============================] - 6s 882us/step - loss: 0.0818 - acc: 0.9648 -
val_loss: 0.3827 - val_acc: 0.9030
Epoch 19/25
7352/7352 [==============================] - 7s 901us/step - loss: 0.0829 - acc: 0.9626 -
val_loss: 0.4073 - val_acc: 0.9053
Epoch 20/25
7352/7352 [==============================] - 7s 909us/step - loss: 0.0736 - acc: 0.9679 -
val_loss: 0.3879 - val_acc: 0.9080
Epoch 21/25
7352/7352 [==============================] - 7s 907us/step - loss: 0.0702 - acc: 0.9702 -
val_loss: 0.3952 - val_acc: 0.9111
Epoch 22/25
7352/7352 [==============================] - 7s 892us/step - loss: 0.0782 - acc: 0.9668 -
val_loss: 0.4166 - val_acc: 0.8836
Epoch 23/25
7352/7352 [==============================] - 7s 907us/step - loss: 0.0759 - acc: 0.9653 -
val_loss: 0.3377 - val_acc: 0.9097
Epoch 24/25
7352/7352 [==============================] - 7s 905us/step - loss: 0.0694 - acc: 0.9680 -
val_loss: 0.3837 - val_acc: 0.9046
Epoch 25/25
7352/7352 [==============================] - 7s 903us/step - loss: 0.0682 - acc: 0.9697 -
val_loss: 0.4052 - val_acc: 0.8955
Time taken :  0:02:46.306998
```

In [29]:

```
score = model.evaluate(X_test, Y_test,verbose=0)
```

In [31]:

```
print("Test Score " , score[0])
print("Test Accuracy " , score[1])
```

```
Test Score  0.40516277702925346
Test Accuracy  0.8954869358669834
```

Model 2

In [40]:

```
epochs = 25
batch_size= 128
```

In [46]:

```
model = Sequential()
model.add(Conv1D(filters=128, kernel_size=5, activation='relu',kernel_initializer='he_uniform',inpu
t_shape=(128,9)))
model.add(Conv1D(filters=64, kernel_size=5, activation='relu',kernel_initializer='he_uniform'))
model.add(Dropout(0.8))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=64, kernel_size=4, activation='relu',kernel_initializer='he_uniform'))
model.add(Dropout(0.8))
model.add(Flatten())
model.add(Dense(50, activation='relu'))
model.add(Dense(6, activation='softmax'))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_10 (Conv1D)           (None, 124, 128)          5888
_________________________________________________________________
conv1d_11 (Conv1D)           (None, 120, 64)           41024
_________________________________________________________________
dropout_6 (Dropout)          (None, 120, 64)           0
_________________________________________________________________
max_pooling1d_5 (MaxPooling1 (None, 60, 64)            0
_________________________________________________________________
conv1d_12 (Conv1D)           (None, 57, 64)            16448
_________________________________________________________________
dropout_7 (Dropout)          (None, 57, 64)            0
_________________________________________________________________
flatten_5 (Flatten)          (None, 3648)              0
_________________________________________________________________
dense_9 (Dense)              (None, 50)                182450
_________________________________________________________________
dense_10 (Dense)             (None, 6)                 306
=================================================================
Total params: 246,116
Trainable params: 246,116
Non-trainable params: 0
_________________________________________________________________
```

In [47]:

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

In [48]:

```
import warnings
warnings.filterwarnings("ignore")

from datetime import datetime
start = datetime.now()

# Training the model
```

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

print("Time taken : ", datetime.now() - start)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/25
7352/7352 [==============================] - 28s 4ms/step - loss: 1.5818 - acc: 0.3770 - val_loss:
1.0215 - val_acc: 0.6427
Epoch 2/25
7352/7352 [==============================] - 28s 4ms/step - loss: 0.6450 - acc: 0.7067 - val_loss:
0.7343 - val_acc: 0.6356
Epoch 3/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.4303 - acc: 0.8186 - val_loss:
0.5463 - val_acc: 0.8127
Epoch 4/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.2856 - acc: 0.8857 - val_loss:
0.5511 - val_acc: 0.8045
Epoch 5/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.2004 - acc: 0.9208 - val_loss:
0.4848 - val_acc: 0.8415
Epoch 6/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1779 - acc: 0.9301 - val_loss:
0.4217 - val_acc: 0.8856
Epoch 7/25
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1595 - acc: 0.9392 - val_loss:
0.4178 - val_acc: 0.8853
Epoch 8/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1364 - acc: 0.9453 - val_loss:
0.4497 - val_acc: 0.8707
Epoch 9/25
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1316 - acc: 0.9455 - val_loss:
0.3743 - val_acc: 0.8911
Epoch 10/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1248 - acc: 0.9461 - val_loss:
0.3724 - val_acc: 0.8985
Epoch 11/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1199 - acc: 0.9474 - val_loss:
0.3797 - val_acc: 0.8853
Epoch 12/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1164 - acc: 0.9509 - val_loss:
0.4762 - val_acc: 0.8361
Epoch 13/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1332 - acc: 0.9396 - val_loss:
0.3518 - val_acc: 0.9030
Epoch 14/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1110 - acc: 0.9513 - val_loss:
0.3452 - val_acc: 0.9070
Epoch 15/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1077 - acc: 0.9538 - val_loss:
0.3818 - val_acc: 0.9060
Epoch 16/25
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1082 - acc: 0.9540 - val_loss:
0.3839 - val_acc: 0.8951
Epoch 17/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1056 - acc: 0.9536 - val_loss:
0.3387 - val_acc: 0.9097
Epoch 18/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1286 - acc: 0.9498 - val_loss:
0.3939 - val_acc: 0.8867
Epoch 19/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1270 - acc: 0.9516 - val_loss:
0.3276 - val_acc: 0.9155
Epoch 20/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.1156 - acc: 0.9510 - val_loss:
0.3560 - val_acc: 0.8921
Epoch 21/25
7352/7352 [==============================] - 28s 4ms/step - loss: 0.1015 - acc: 0.9551 - val_loss:
0.3366 - val_acc: 0.9111
Epoch 22/25
7352/7352 [==============================] - 29s 4ms/step - loss: 0.0992 - acc: 0.9540 - val_loss:
0.3152 - val_acc: 0.9179
Epoch 23/25
```

```
7352/7352 [==============================] - 29s 4ms/step - loss: 0.0966 - acc: 0.9561 - val_loss:
0.3435 - val_acc: 0.9026
Epoch 24/25
7352/7352 [==============================] - 29s 4ms/step - loss: 0.0967 - acc: 0.9569 - val_loss:
0.3432 - val_acc: 0.9196
Epoch 25/25
7352/7352 [==============================] - 27s 4ms/step - loss: 0.0872 - acc: 0.9606 - val_loss:
0.3525 - val_acc: 0.9169
Time taken :  0:11:25.867490
```

In [49]:

```
score = model.evaluate(X_test, Y_test,verbose=0)
```

In [50]:

```
print("Test Score " , score[0])
print("Test Accuracy " , score[1])
```

```
Test Score  0.3524832843419216
Test Accuracy  0.9168646080760094
```

Model 3

In [51]:

```
epochs = 40
batch_size= 128
```

In [52]:

```
model = Sequential()
model.add(Conv1D(filters=246, kernel_size=6, activation='relu',kernel_initializer='random_uniform',
input_shape=(128,9)))
model.add(Conv1D(filters=128, kernel_size=5, activation='relu',kernel_initializer='random_uniform')
)
model.add(Dropout(0.88))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=64, kernel_size=6, activation='relu',kernel_initializer='he_uniform'))
model.add(Dropout(0.89))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dense(6, activation='softmax'))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_13 (Conv1D)           (None, 123, 246)          13530
_________________________________________________________________
conv1d_14 (Conv1D)           (None, 119, 128)          157568
_________________________________________________________________
dropout_8 (Dropout)          (None, 119, 128)          0
_________________________________________________________________
max_pooling1d_6 (MaxPooling1 (None, 59, 128)           0
_________________________________________________________________
conv1d_15 (Conv1D)           (None, 54, 64)            49216
_________________________________________________________________
dropout_9 (Dropout)          (None, 54, 64)            0
_________________________________________________________________
flatten_6 (Flatten)          (None, 3456)              0
_________________________________________________________________
dense_11 (Dense)             (None, 64)                221248
_________________________________________________________________
dense_12 (Dense)             (None, 6)                 390
=================================================================
Total params: 441,952
Trainable params: 441,952
Non-trainable params: 0
_________________________________________________________________
```

In [53]:

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

In [54]:

```
import warnings
warnings.filterwarnings("ignore")

from datetime import datetime
start = datetime.now()

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

print("Time taken : ", datetime.now() - start)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/40
7352/7352 [==============================] - 64s 9ms/step - loss: 1.1683 - acc: 0.4629 - val_loss:
0.8387 - val_acc: 0.6624
Epoch 2/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.5551 - acc: 0.7514 - val_loss:
0.6162 - val_acc: 0.7978
Epoch 3/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2940 - acc: 0.8890 - val_loss:
0.4443 - val_acc: 0.8907
Epoch 4/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1718 - acc: 0.9342 - val_loss:
0.4105 - val_acc: 0.8972
Epoch 5/40
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1437 - acc: 0.9421 - val_loss:
0.5445 - val_acc: 0.8833
Epoch 6/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1345 - acc: 0.9425 - val_loss:
0.4486 - val_acc: 0.9033
Epoch 7/40
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1231 - acc: 0.9490 - val_loss:
0.2757 - val_acc: 0.9175
Epoch 8/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1351 - acc: 0.9482 - val_loss:
0.2920 - val_acc: 0.9294
Epoch 9/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1989 - acc: 0.9440 - val_loss:
0.3191 - val_acc: 0.9220
Epoch 10/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2332 - acc: 0.9446 - val_loss:
0.3474 - val_acc: 0.9165
Epoch 11/40
7352/7352 [==============================] - 64s 9ms/step - loss: 0.2224 - acc: 0.9480 - val_loss:
0.3186 - val_acc: 0.9281
Epoch 12/40
7352/7352 [==============================] - 64s 9ms/step - loss: 0.2189 - acc: 0.9480 - val_loss:
0.3465 - val_acc: 0.9084
Epoch 13/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2245 - acc: 0.9444 - val_loss:
0.3633 - val_acc: 0.9247
Epoch 14/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2130 - acc: 0.9480 - val_loss:
0.3928 - val_acc: 0.9342
Epoch 15/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2102 - acc: 0.9480 - val_loss:
0.4068 - val_acc: 0.9230
Epoch 16/40
7352/7352 [==============================] - 64s 9ms/step - loss: 0.2158 - acc: 0.9475 - val_loss:
0.4732 - val_acc: 0.9152
Epoch 17/40
7352/7352 [==============================] - 64s 9ms/step - loss: 0.3718 - acc: 0.9241 - val_loss:
0.4865 - val_acc: 0.8839
Epoch 18/40
```

```
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2894 - acc: 0.9343 - val_loss:
0.5460 - val_acc: 0.9036
Epoch 19/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2736 - acc: 0.9411 - val_loss:
0.5387 - val_acc: 0.9121
Epoch 20/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2329 - acc: 0.9421 - val_loss:
0.4620 - val_acc: 0.9203
Epoch 21/40
7352/7352 [==============================] - 64s 9ms/step - loss: 0.2325 - acc: 0.9426 - val_loss:
0.5261 - val_acc: 0.9155
Epoch 22/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2760 - acc: 0.9391 - val_loss:
0.5391 - val_acc: 0.9179
Epoch 23/40
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1747 - acc: 0.9490 - val_loss:
0.5266 - val_acc: 0.9165
Epoch 24/40
7352/7352 [==============================] - 65s 9ms/step - loss: 0.1451 - acc: 0.9524 - val_loss:
0.5046 - val_acc: 0.9148
Epoch 25/40
7352/7352 [==============================] - 67s 9ms/step - loss: 0.1696 - acc: 0.9508 - val_loss:
0.3547 - val_acc: 0.9220
Epoch 26/40
7352/7352 [==============================] - 66s 9ms/step - loss: 0.1570 - acc: 0.9484 - val_loss:
0.4063 - val_acc: 0.9223
Epoch 27/40
7352/7352 [==============================] - 66s 9ms/step - loss: 0.3695 - acc: 0.9312 - val_loss:
0.6576 - val_acc: 0.8707
Epoch 28/40
7352/7352 [==============================] - 65s 9ms/step - loss: 0.3114 - acc: 0.9366 - val_loss:
0.3791 - val_acc: 0.9186
Epoch 29/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2377 - acc: 0.9455 - val_loss:
0.4866 - val_acc: 0.9128
Epoch 30/40
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2316 - acc: 0.9504 - val_loss:
0.4326 - val_acc: 0.9114
Epoch 31/40
7352/7352 [==============================] - 65s 9ms/step - loss: 0.2582 - acc: 0.9468 - val_loss:
0.3488 - val_acc: 0.9315
Epoch 32/40
7352/7352 [==============================] - 69s 9ms/step - loss: 0.2443 - acc: 0.9494 - val_loss:
0.3880 - val_acc: 0.9250
Epoch 33/40
7352/7352 [==============================] - 67s 9ms/step - loss: 0.2383 - acc: 0.9476 - val_loss:
0.3674 - val_acc: 0.9233
Epoch 34/40
7352/7352 [==============================] - 68s 9ms/step - loss: 0.2317 - acc: 0.9504 - val_loss:
0.3841 - val_acc: 0.9209
Epoch 35/40
7352/7352 [==============================] - 68s 9ms/step - loss: 0.2029 - acc: 0.9532 - val_loss:
0.4248 - val_acc: 0.9192
Epoch 36/40
7352/7352 [==============================] - 67s 9ms/step - loss: 0.2109 - acc: 0.9521 - val_loss:
0.5454 - val_acc: 0.9216
Epoch 37/40
7352/7352 [==============================] - 65s 9ms/step - loss: 0.2366 - acc: 0.9502 - val_loss:
0.4279 - val_acc: 0.9216
Epoch 38/40
7352/7352 [==============================] - 67s 9ms/step - loss: 0.2220 - acc: 0.9551 - val_loss:
0.4350 - val_acc: 0.9192
Epoch 39/40
7352/7352 [==============================] - 66s 9ms/step - loss: 0.2051 - acc: 0.9548 - val_loss:
0.4016 - val_acc: 0.9199
Epoch 40/40
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1992 - acc: 0.9551 - val_loss:
0.4385 - val_acc: 0.9216
Time taken :  0:43:00.707163
```

In [55]:

```
score = model.evaluate(X_test, Y_test,verbose=0)
```

In [56]:

In [56]:

```
print("Test Score " , score[0])
print("Test Accuracy " , score[1])
```

```
Test Score  0.4384836606237519
Test Accuracy  0.9216152019002375
```

Model 4

In [57]:

```
epochs = 30
batch_size= 128
```

In [59]:

```
model = Sequential()
model.add(Conv1D(filters=512, kernel_size=6, activation='relu',kernel_initializer='random_uniform',
input_shape=(128,9)))
model.add(Conv1D(filters=256, kernel_size=5, activation='relu',kernel_initializer='random_uniform')
)
model.add(Dropout(0.88))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=128, kernel_size=6, activation='relu',kernel_initializer='random_uniform')
)
model.add(Dropout(0.9))
model.add(Conv1D(filters=128, kernel_size=6, activation='relu',kernel_initializer='random_uniform')
)
model.add(Dropout(0.9))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dense(6, activation='softmax'))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_20 (Conv1D)           (None, 123, 512)          28160
_________________________________________________________________
conv1d_21 (Conv1D)           (None, 119, 256)          655616
_________________________________________________________________
dropout_12 (Dropout)         (None, 119, 256)          0
_________________________________________________________________
max_pooling1d_8 (MaxPooling1 (None, 59, 256)           0
_________________________________________________________________
conv1d_22 (Conv1D)           (None, 54, 128)           196736
_________________________________________________________________
dropout_13 (Dropout)         (None, 54, 128)           0
_________________________________________________________________
conv1d_23 (Conv1D)           (None, 49, 128)           98432
_________________________________________________________________
dropout_14 (Dropout)         (None, 49, 128)           0
_________________________________________________________________
flatten_8 (Flatten)          (None, 6272)              0
_________________________________________________________________
dense_15 (Dense)             (None, 64)                401472
_________________________________________________________________
dense_16 (Dense)             (None, 6)                 390
=================================================================
Total params: 1,380,806
Trainable params: 1,380,806
Non-trainable params: 0
_________________________________________________________________
```

In [60]:

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

In [61]:

```
import warnings
warnings.filterwarnings("ignore")

from datetime import datetime
start = datetime.now()

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

print("Time taken : ", datetime.now() - start)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 215s 29ms/step - loss: 12.0574 - acc: 0.1881 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 2/30
7352/7352 [==============================] - 213s 29ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 3/30
7352/7352 [==============================] - 213s 29ms/step - loss: 13.0240 - acc: 0.1919 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 4/30
7352/7352 [==============================] - 213s 29ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 5/30
7352/7352 [==============================] - 213s 29ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 6/30
7352/7352 [==============================] - 217s 29ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 7/30
7352/7352 [==============================] - 222s 30ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 8/30
7352/7352 [==============================] - 222s 30ms/step - loss: 13.0357 - acc: 0.1912 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 9/30
7352/7352 [==============================] - 227s 31ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 10/30
7352/7352 [==============================] - 219s 30ms/step - loss: 13.0313 - acc: 0.1915 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 11/30
7352/7352 [==============================] - 216s 29ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 12/30
7352/7352 [==============================] - 217s 29ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 13/30
7352/7352 [==============================] - 217s 29ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 14/30
7352/7352 [==============================] - 224s 30ms/step - loss: 13.0291 - acc: 0.1916 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 15/30
7352/7352 [==============================] - 227s 31ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 16/30
7352/7352 [==============================] - 223s 30ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 17/30
7352/7352 [==============================] - 138s 19ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 18/30
7352/7352 [==============================] - 104s 14ms/step - loss: 13.1626 - acc: 0.1832 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 19/30
7352/7352 [==============================] - 105s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 20/30
7352/7352 [==============================] - 104s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 21/30
```

```
Epoch 21/30
7352/7352 [==============================] - 104s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 22/30
7352/7352 [==============================] - 104s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 23/30
7352/7352 [==============================] - 105s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 24/30
7352/7352 [==============================] - 104s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 25/30
7352/7352 [==============================] - 104s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 26/30
7352/7352 [==============================] - 104s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 27/30
7352/7352 [==============================] - 104s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 28/30
7352/7352 [==============================] - 104s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 29/30
7352/7352 [==============================] - 104s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Epoch 30/30
7352/7352 [==============================] - 103s 14ms/step - loss: 13.0335 - acc: 0.1914 - val_lo
ss: 13.1811 - val_acc: 0.1822
Time taken :  1:23:07.805060
```

In [62]:

```
score = model.evaluate(X_test, Y_test,verbose=0)
```

In [63]:

```
print("Test Score " , score[0])
print("Test Accuracy " , score[1])
```

```
Test Score  13.181068860517748
Test Accuracy  0.18221920597217509
```

Prettytable for layer 1

In [66]:

```
number       = [1, 2, 3, 4]
epochs       = [35, 30, 30,25]
batch_size   = [32, 32, 32,128]
n_hidden     = [256, 512, 128,512]
drop_out     = [0.65, 0.80, 0.50,0.80]
accuracy     = [90.60, 16.84, 92.60, 86.73]

# Initializing prettytable
ptable = PrettyTable()
ptable.add_column("Example",number)
ptable.add_column("Epochs", epochs)
ptable.add_column("Batch Size",batch_size)
ptable.add_column("Hidden Layer",n_hidden)
ptable.add_column("Dropout",drop_out)
ptable.add_column("Accuracy %",accuracy)
print(ptable)
```

```
+---------+--------+------------+--------------+---------+------------+
| Example | Epochs | Batch Size | Hidden Layer | Dropout | Accuracy % |
+---------+--------+------------+--------------+---------+------------+
|    1    |   35   |     32     |     256      |   0.65  |    90.6    |
|    2    |   30   |     32     |     512      |   0.8   |   16.84    |
|    3    |   30   |     32     |     128      |   0.5   |    92.6    |
|    4    |   25   |    128     |     512      |   0.8   |   86.73    |
```

```
+---------+--------+------------+--------------+---------+------------+
```

Prettytable for layer 2

In [58]:

```
number          = [1, 2]
epochs          = [30, 50]
batch_size      = [32, 64]
n_hidden_layer1 = [128, 32]
n_hidden_layer2 = [64, 64]
drop_out_1      = [0.2, 0.5]
drop_out_2      = [0.5, 0.5]
accuracy     = [92.43, 91.52]

# Initializing prettytable
ptable = PrettyTable()
ptable.add_column("Example",number)
ptable.add_column("Epochs", epochs)
ptable.add_column("Batch Size",batch_size)
ptable.add_column("Hidden Layer 1",n_hidden_layer1)
ptable.add_column("Hidden Layer 2",n_hidden_layer2)
ptable.add_column("Dropout 1",drop_out_1)
ptable.add_column("Dropout 2",drop_out_2)
ptable.add_column("Accuracy %",accuracy)
print(ptable)
```

```
+---------+--------+------------+----------------+----------------+-----------+-----------+--------
-+
| Example | Epochs | Batch Size | Hidden Layer 1 | Hidden Layer 2 | Dropout 1 | Dropout 2 | Accura
cy % |
+---------+--------+------------+----------------+----------------+-----------+-----------+--------
-+
|    1    |   30   |     32     |      128       |       64       |    0.2    |    0.5    |   92.43
|
|    2    |   50   |     64     |       32       |       64       |    0.5    |    0.5    |   91.52
|
+---------+--------+------------+----------------+----------------+-----------+-----------+--------
-+
```

Prettytable for CNN

In [65]:

```
number          = [1, 2, 3, 4]
epochs          = [25, 25, 40, 30]
batch_size      = [128, 128, 128, 128]
accuracy     = [89.54,91.68,92.16,18.22]

# Initializing prettytable
ptable = PrettyTable()
ptable.add_column("Model",number)
ptable.add_column("Epochs", epochs)
ptable.add_column("Batch Size",batch_size)
ptable.add_column("Accuracy %",accuracy)
print(ptable)
```

```
+-------+--------+------------+------------+
| Model | Epochs | Batch Size | Accuracy % |
+-------+--------+------------+------------+
|   1   |   25   |    128     |   89.54    |
|   2   |   25   |    128     |   91.68    |
|   3   |   40   |    128     |   92.16    |
|   4   |   30   |    128     |   18.22    |
+-------+--------+------------+------------+
```

Conclusions

1. We have used one LSTM layered model and two LSTM layered model.
2. Used different epochs, batch sizes, dropout layers to find accuracy.
3. Maximum accuracy % in one LSTM layer model is 92.60%
4. We also used larger LSTM units with large dropout rates in one layered model.
5. Accuracy % in two LSTM layers model is 92.43%
6. Earlystopping technique can help to get maximum accuracy of 93.72% in two layered architecture.
7. We have tried all different combinations of epochs, batch sizes and dropout values to find maximum accuracy and approx 93% is best accuracy.
8. With different CNN models, we get best accuracy of 92.16%. We have used different architectures of Conv2D with different kernel sizes.

In [ ]: