In [1]:

```
# Importing Libraries
#https://github.com/UdiBhaskar/Human-Activity-Recognition--Using-Deep-NN/blob/master/Human%20Activity%20Detection.ipynb
```

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 [20]:

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

In [21]:

```
# 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)
```

```
Using TensorFlow backend.
```

In [75]:

```
# Importing libraries
import pickle
from keras.models import Sequential
from keras.layers import LSTM,BatchNormalization
from keras.layers.core import Dense, Dropout
from keras.layers import Flatten
from keras.layers.convolutional import Conv1D
```

```
from keras.layers.convolutional import MaxPooling1D
from keras.utils import to_categorical
from keras import optimizers
from sklearn.preprocessing import StandardScaler
from keras.regularizers import l2
import matplotlib.pyplot as plt
```

In [72]:

```
import numpy as np
import matplotlib.pyplot as plt
import numpy as np
import itertools
def plot_confusion_matrix(cm,
                          target_names,
                          title='Confusion matrix',
                          cmap=None,
                          normalize=True):

    accuracy = np.trace(cm) / np.sum(cm).astype('float')
    misclass = 1 - accuracy

    if cmap is None:
        cmap = plt.get_cmap('Blues')

    plt.figure(figsize=(8, 6))
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()

    if target_names is not None:
        tick_marks = np.arange(len(target_names))
        plt.xticks(tick_marks, target_names, rotation=45)
        plt.yticks(tick_marks, target_names)

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]


    thresh = cm.max() / 1.5 if normalize else cm.max() / 2
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        if normalize:
            plt.text(j, i, "{:0.4f}".format(cm[i, j]),
                     horizontalalignment="center",
                     color="white" if cm[i, j] > thresh else "black")
        else:
            plt.text(j, i, "{:,}".format(cm[i, j]),
                     horizontalalignment="center",
                     color="white" if cm[i, j] > thresh else "black")


    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label\naccuracy={:0.4f}; misclass={:0.4f}'.format(accuracy, misclass))
    plt.show()
```

Model for classifying data into Static and Dynamic activities

In [76]:

```
## Classifying data as 2 class dynamic vs static
##data preparation
def data_scaled_2class():
    """
    Obtain the dataset from multiple files.
    Returns: X_train, X_test, y_train, y_test
    """
    # Data directory
    DATADIR = 'UCI_HAR_Dataset'
    # 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"
        ]
    from sklearn.base import BaseEstimator, TransformerMixin
    class scaling_tseries_data(BaseEstimator, TransformerMixin):
        from sklearn.preprocessing import StandardScaler
        def __init__(self):
            self.scale = None

        def transform(self, X):
            temp_X1 = X.reshape((X.shape[0] * X.shape[1], X.shape[2]))
            temp_X1 = self.scale.transform(temp_X1)
            return temp_X1.reshape(X.shape)

        def fit(self, X):
            # remove overlaping
            remove = int(X.shape[1] / 2)
            temp_X = X[:, -remove:, :]
            # flatten data
            temp_X = temp_X.reshape((temp_X.shape[0] * temp_X.shape[1], temp_X.shape[2]))
            scale = StandardScaler()
            scale.fit(temp_X)
            ##saving for furter usage
            ## will use in predicton pipeline
            pickle.dump(scale,open('Scale_2class.p','wb'))
            self.scale = scale
            return self

    # 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))

    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]
        y[y<=3] = 0
        y[y>3] = 1
        return pd.get_dummies(y).as_matrix()

    X_train_2c, X_test_2c = load_signals('train'), load_signals('test')
    Y_train_2c, Y_test_2c = load_y('train'), load_y('test')
    ###Scling data
    Scale = scaling_tseries_data()
    Scale.fit(X_train_2c)
    X_train_2c = Scale.transform(X_train_2c)
    X_test_2c = Scale.transform(X_test_2c)
    return X_train_2c, Y_train_2c, X_test_2c,  Y_test_2c
```

In [77]:

```
X_train_2c, Y_train_2c, X_test_2c,  Y_test_2c = data_scaled_2class()
print(Y_train_2c.shape)
print(Y_test_2c.shape)
```

```
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-
packages\ipykernel_launcher.py:62: FutureWarning: Method .as_matrix will be removed in a future ve
rsion. Use .values instead.
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-
packages\ipykernel_launcher.py:80: FutureWarning: Method .as_matrix will be removed in a future ve
rsion. Use .values instead.
```

```
(7352, 2)
(2947, 2)
```

In [78]:

```
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(2, activation='softmax'))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv1d_11 (Conv1D)           (None, 126, 32)           896       
_________________________________________________________________
conv1d_12 (Conv1D)           (None, 124, 32)           3104      
_________________________________________________________________
dropout_6 (Dropout)          (None, 124, 32)           0         
_________________________________________________________________
max_pooling1d_6 (MaxPooling1 (None, 62, 32)            0         
_________________________________________________________________
flatten_6 (Flatten)          (None, 1984)              0         
_________________________________________________________________
dense_11 (Dense)             (None, 50)                99250     
_________________________________________________________________
dense_12 (Dense)             (None, 2)                 102       
=================================================================
Total params: 103,352
Trainable params: 103,352
Non-trainable params: 0
_________________________________________________________________
```

In [79]:

```
model.compile(loss='categorical_crossentropy', optimizer=optimizers.Adam(lr=0.001),
metrics=['accuracy'])
model.fit(X_train_2c,Y_train_2c, epochs=20, batch_size=16,validation_data=(X_test_2c, Y_test_2c), v
erbose=1)
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/20
7352/7352 [==============================] - 5s 719us/step - loss: 0.0386 - acc: 0.9875 -
val_loss: 0.0362 - val_acc: 0.9793
Epoch 2/20
7352/7352 [==============================] - 5s 715us/step - loss: 0.0045 - acc: 0.9988 -
val_loss: 0.0149 - val_acc: 0.9925
Epoch 3/20
7352/7352 [==============================] - 5s 653us/step - loss: 0.0024 - acc: 0.9992 -
val_loss: 0.0147 - val_acc: 0.9969
Epoch 4/20
7352/7352 [==============================] - 5s 625us/step - loss: 0.0022 - acc: 0.9996 -
val_loss: 0.0332 - val_acc: 0.9902
Epoch 5/20
7352/7352 [==============================] - 4s 601us/step - loss: 1.9323e-06 - acc: 1.0000 - val
```

```
loss: 0.0334 - val_acc: 0.9902
Epoch 6/20
7352/7352 [==============================] - 5s 635us/step - loss: 3.1750e-06 - acc: 1.0000 - val_
loss: 0.0312 - val_acc: 0.9908
Epoch 7/20
7352/7352 [==============================] - 5s 614us/step - loss: 4.7979e-05 - acc: 1.0000 - val_
loss: 0.0177 - val_acc: 0.9925
Epoch 8/20
7352/7352 [==============================] - 5s 681us/step - loss: 3.7395e-07 - acc: 1.0000 - val_
loss: 0.0180 - val_acc: 0.9925
Epoch 9/20
7352/7352 [==============================] - 5s 656us/step - loss: 3.1331e-07 - acc: 1.0000 - val_
loss: 0.0182 - val_acc: 0.9925
Epoch 10/20
7352/7352 [==============================] - 5s 656us/step - loss: 7.0031e-07 - acc: 1.0000 - val_
loss: 0.0192 - val_acc: 0.9922
Epoch 11/20
7352/7352 [==============================] - 5s 710us/step - loss: 3.4709e-07 - acc: 1.0000 - val_
loss: 0.0199 - val_acc: 0.9922
Epoch 12/20
7352/7352 [==============================] - 5s 734us/step - loss: 4.0008e-07 - acc: 1.0000 - val_
loss: 0.0197 - val_acc: 0.9922
Epoch 13/20
7352/7352 [==============================] - 5s 708us/step - loss: 6.3465e-07 - acc: 1.0000 - val_
loss: 0.0203 - val_acc: 0.9922
Epoch 14/20
7352/7352 [==============================] - 5s 692us/step - loss: 2.1935e-07 - acc: 1.0000 - val_
loss: 0.0202 - val_acc: 0.9922
Epoch 15/20
7352/7352 [==============================] - 5s 632us/step - loss: 1.1870e-06 - acc: 1.0000 - val_
loss: 0.0133 - val_acc: 0.9980
Epoch 16/20
7352/7352 [==============================] - 5s 663us/step - loss: 7.9860e-07 - acc: 1.0000 - val_
loss: 0.0178 - val_acc: 0.9925
Epoch 17/20
7352/7352 [==============================] - 4s 582us/step - loss: 2.0351e-07 - acc: 1.0000 - val_
loss: 0.0179 - val_acc: 0.9925
Epoch 18/20
7352/7352 [==============================] - 4s 574us/step - loss: 2.6607e-07 - acc: 1.0000 - val_
loss: 0.0167 - val_acc: 0.9929
Epoch 19/20
7352/7352 [==============================] - 4s 582us/step - loss: 1.3941e-07 - acc: 1.0000 - val_
loss: 0.0170 - val_acc: 0.9925
Epoch 20/20
7352/7352 [==============================] - 4s 580us/step - loss: 2.4786e-07 - acc: 1.0000 - val_
loss: 0.0204 - val_acc: 0.9919
```

Out[79]:

```
<keras.callbacks.History at 0x1a028790898>
```

In [80]:

```
_,acc_test = model.evaluate(X_test_2c,Y_test_2c,verbose=0)
_,acc_train = model.evaluate(X_train_2c,Y_train_2c,verbose=0)
print('Train_accuracy',acc_train,'test_accuracy',acc_test)
```

```
Train_accuracy 1.0 test_accuracy 0.991856124872752
```

In [81]:

```
##saving model
model.save('final_model_2class.h5')
```

## Classificaton of Static activities

In [82]:

```
def data_scaled_static():
    """
    Obtain the dataset from multiple files.
```

```
    Returns: X_train, X_test, y_train, y_test
    """
    # Data directory
    DATADIR = 'UCI_HAR_Dataset'
    # 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"
        ]
    from sklearn.base import BaseEstimator, TransformerMixin
    class scaling_tseries_data(BaseEstimator, TransformerMixin):
        from sklearn.preprocessing import StandardScaler
        def __init__(self):
            self.scale = None

        def transform(self, X):
            temp_X1 = X.reshape((X.shape[0] * X.shape[1], X.shape[2]))
            temp_X1 = self.scale.transform(temp_X1)
            return temp_X1.reshape(X.shape)

        def fit(self, X):
            # remove overlaping
            remove = int(X.shape[1] / 2)
            temp_X = X[:, -remove:, :]
            # flatten data
            temp_X = temp_X.reshape((temp_X.shape[0] * temp_X.shape[1], temp_X.shape[2]))
            scale = StandardScaler()
            scale.fit(temp_X)
            #for furter use at prediction pipeline
            pickle.dump(scale,open('Scale_static.p','wb'))
            self.scale = scale
            return self

    # 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))

    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]
        y_subset = y>3
        y = y[y_subset]
        return pd.get_dummies(y).as_matrix(),y_subset

    Y_train_s,y_train_sub = load_y('train')
    Y_test_s,y_test_sub = load_y('test')
```

```
    X_train_s, X_test_s = load_signals('train'), load_signals('test')
    X_train_s = X_train_s[y_train_sub]
    X_test_s = X_test_s[y_test_sub]

    ###Scling data
    Scale = scaling_tseries_data()
    Scale.fit(X_train_s)
    X_train_s = Scale.transform(X_train_s)
    X_test_s = Scale.transform(X_test_s)

    return X_train_s, Y_train_s, X_test_s,  Y_test_s
```

In [83]:

```
X_train_s, Y_train_s, X_test_s,  Y_test_s = data_scaled_static()
```

```
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-
packages\ipykernel_launcher.py:77: FutureWarning: Method .as_matrix will be removed in a future ve
rsion. Use .values instead.
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-
packages\ipykernel_launcher.py:59: FutureWarning: Method .as_matrix will be removed in a future ve
rsion. Use .values instead.
```

In [84]:

```
print('X Shape of train data',X_train_s.shape, 'Y shape', Y_train_s.shape)
print('X Shape of val data',X_test_s.shape,'Y shape',Y_test_s.shape)
```

```
X Shape of train data (4067, 128, 9) Y shape (4067, 3)
X Shape of val data (1560, 128, 9) Y shape (1560, 3)
```

In [85]:

```
def keras_fmin_fnct(space,verbose=1):
    np.random.seed(0)
    tf.set_random_seed(0)
    sess = tf.Session(graph=tf.get_default_graph())
    K.set_session(sess)
    # Initiliazing the sequential model
    model = Sequential()
    model.add(Conv1D(filters=space['filters'], kernel_size=space['kernel_size'],activation='relu',
                    kernel_initializer='he_uniform',
                    kernel_regularizer=l2(space['l2']),input_shape=(128,9)))
    model.add(Conv1D(filters=space['filters_1'], kernel_size=space['kernel_size_1'],

activation='relu',kernel_regularizer=l2(space['l2_1']),kernel_initializer='he_uniform'))
    model.add(Dropout(space['Dropout']))
    model.add(MaxPooling1D(pool_size=space['pool_size']))
    model.add(Flatten())
    model.add(Dense(space['Dense'], activation='relu'))
    model.add(Dense(3, activation='softmax'))
    adam = optimizers.Adam(lr=space['lr'])
    rmsprop = optimizers.RMSprop(lr=space['lr_1'])
    choiceval = space['choiceval']
    if choiceval == 'adam':
        optim = adam
    else:
        optim = rmsprop
    print(model.summary())
    model.compile(loss='categorical_crossentropy', metrics=['accuracy'],optimizer=optim)
    result = model.fit(X_train_s, Y_train_s,
                    batch_size=space['Dense_1'],
                    nb_epoch=space['nb_epoch'],
                    verbose=verbose,
                    validation_data=(X_test_s, Y_test_s))
    #K.clear_session()
    return model,result
```

In [86]:

```
params_1={'Dense': 64,
 'Dense_1': 64,
```

```
 'Dense_1': 64,
 'Dropout': 0.45377377480700615,
 'choiceval': 'rmsprop',
 'filters': 32,
 'filters_1': 16,
 'kernel_size': 5,
 'kernel_size_1': 3,
 'l2': 0.0019801221163149862,
 'l2_1': 0.8236255110533577,
 'lr': 0.003918784585237195,
 'lr_1': 0.002237071747066137,
 'nb_epoch': 30,
 'pool_size': 2}
best_model,result = keras_fmin_fnct(params_1)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv1d_13 (Conv1D)           (None, 124, 32)           1472      
_________________________________________________________________
conv1d_14 (Conv1D)           (None, 122, 16)           1552      
_________________________________________________________________
dropout_7 (Dropout)          (None, 122, 16)           0         
_________________________________________________________________
max_pooling1d_7 (MaxPooling1 (None, 61, 16)            0         
_________________________________________________________________
flatten_7 (Flatten)          (None, 976)               0         
_________________________________________________________________
dense_13 (Dense)             (None, 64)                62528     
_________________________________________________________________
dense_14 (Dense)             (None, 3)                 195       
=================================================================
Total params: 65,747
Trainable params: 65,747
Non-trainable params: 0
_________________________________________________________________
None
```

```
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-
packages\ipykernel_launcher.py:31: UserWarning: The `nb_epoch` argument in `fit` has been renamed
`epochs`.
```

```
Train on 4067 samples, validate on 1560 samples
Epoch 1/30
4067/4067 [==============================] - 2s 437us/step - loss: 10.6870 - acc: 0.8333 -
val_loss: 3.0266 - val_acc: 0.9006
Epoch 2/30
4067/4067 [==============================] - 2s 388us/step - loss: 1.3128 - acc: 0.8955 -
val_loss: 0.6611 - val_acc: 0.8705
Epoch 3/30
4067/4067 [==============================] - 1s 353us/step - loss: 0.4724 - acc: 0.9036 -
val_loss: 0.5028 - val_acc: 0.8526
Epoch 4/30
4067/4067 [==============================] - 1s 357us/step - loss: 0.3897 - acc: 0.9110 -
val_loss: 0.4572 - val_acc: 0.8538
Epoch 5/30
4067/4067 [==============================] - 2s 400us/step - loss: 0.3501 - acc: 0.9137 -
val_loss: 0.4066 - val_acc: 0.8795
Epoch 6/30
4067/4067 [==============================] - 2s 381us/step - loss: 0.3264 - acc: 0.9166 -
val_loss: 0.3806 - val_acc: 0.8705
Epoch 7/30
4067/4067 [==============================] - 1s 366us/step - loss: 0.3255 - acc: 0.9147 -
val_loss: 0.6140 - val_acc: 0.8571
Epoch 8/30
4067/4067 [==============================] - 1s 342us/step - loss: 0.2770 - acc: 0.9289 -
val_loss: 0.4432 - val_acc: 0.8628
Epoch 9/30
4067/4067 [==============================] - 1s 362us/step - loss: 0.2774 - acc: 0.9275 -
val_loss: 0.2983 - val_acc: 0.9186
Epoch 10/30
4067/4067 [==============================] - 1s 335us/step - loss: 0.2567 - acc: 0.9294 -
val_loss: 0.2868 - val_acc: 0.9071
Epoch 11/30
4067/4067 [==============================] - 2s 375us/step - loss: 0.2484 - acc: 0.9321 -
val_loss: 0.3229 - val_acc: 0.8846
```

```
Epoch 12/30
4067/4067 [==============================] - 2s 392us/step - loss: 0.2468 - acc: 0.9331 -
val_loss: 0.2940 - val_acc: 0.9205
Epoch 13/30
4067/4067 [==============================] - 2s 395us/step - loss: 0.2329 - acc: 0.9375 -
val_loss: 0.3378 - val_acc: 0.8891
Epoch 14/30
4067/4067 [==============================] - 2s 413us/step - loss: 0.2340 - acc: 0.9398 -
val_loss: 0.2612 - val_acc: 0.9308
Epoch 15/30
4067/4067 [==============================] - 2s 418us/step - loss: 0.2185 - acc: 0.9412 -
val_loss: 0.2716 - val_acc: 0.9019
Epoch 16/30
4067/4067 [==============================] - 2s 415us/step - loss: 0.2062 - acc: 0.9425 -
val_loss: 0.3121 - val_acc: 0.8885
Epoch 17/30
4067/4067 [==============================] - 1s 367us/step - loss: 0.1973 - acc: 0.9469 -
val_loss: 0.2888 - val_acc: 0.8987
Epoch 18/30
4067/4067 [==============================] - 2s 420us/step - loss: 0.2661 - acc: 0.9403 -
val_loss: 0.3274 - val_acc: 0.8865
Epoch 19/30
4067/4067 [==============================] - 2s 400us/step - loss: 0.2099 - acc: 0.9412 -
val_loss: 0.2674 - val_acc: 0.9090
Epoch 20/30
4067/4067 [==============================] - 1s 358us/step - loss: 0.1971 - acc: 0.9466 -
val_loss: 0.2634 - val_acc: 0.9058
Epoch 21/30
4067/4067 [==============================] - 2s 378us/step - loss: 0.1965 - acc: 0.9459 -
val_loss: 0.2611 - val_acc: 0.9103
Epoch 22/30
4067/4067 [==============================] - 2s 379us/step - loss: 0.1833 - acc: 0.9506 -
val_loss: 0.2600 - val_acc: 0.9135
Epoch 23/30
4067/4067 [==============================] - 2s 381us/step - loss: 0.2018 - acc: 0.9452 -
val_loss: 0.2637 - val_acc: 0.9058
Epoch 24/30
4067/4067 [==============================] - 2s 370us/step - loss: 0.1962 - acc: 0.9479 -
val_loss: 0.2524 - val_acc: 0.9167 - loss: 0.1938 - acc: 0.949
Epoch 25/30
4067/4067 [==============================] - 1s 360us/step - loss: 0.1920 - acc: 0.9469 -
val_loss: 0.2976 - val_acc: 0.8968
Epoch 26/30
4067/4067 [==============================] - 2s 373us/step - loss: 0.1858 - acc: 0.9498 -
val_loss: 0.2361 - val_acc: 0.9410
Epoch 27/30
4067/4067 [==============================] - 2s 374us/step - loss: 0.1799 - acc: 0.9501 -
val_loss: 0.2628 - val_acc: 0.9071
Epoch 28/30
4067/4067 [==============================] - 2s 415us/step - loss: 0.1832 - acc: 0.9560 -
val_loss: 0.2364 - val_acc: 0.9391
Epoch 29/30
4067/4067 [==============================] - 1s 368us/step - loss: 0.1939 - acc: 0.9548 -
val_loss: 0.2562 - val_acc: 0.9314
Epoch 30/30
4067/4067 [==============================] - 2s 398us/step - loss: 0.1749 - acc: 0.9577 -
val_loss: 0.2773 - val_acc: 0.9122
```

In [87]:

```
_,acc_test = best_model.evaluate(X_test_s,Y_test_s,verbose=0)
_,acc_train = best_model.evaluate(X_train_s,Y_train_s,verbose=0)
print('Train_accuracy',acc_train,'test_accuracy',acc_test)
```

```
Train_accuracy 0.9589377919842635 test_accuracy 0.9121794871794872
```

In [88]:

```
params_2={'Dense': 64,
 'Dense_1': 64,
 'Dropout': 0.45377377480700615,
 'choiceval': 'rmsprop',
 'filters': 32,
 'filters_1': 16,
```

```
 'kernel_size': 5,
 'kernel_size_1': 3,
 'l2': 0.0019801221163149862,
 'l2_1': 0.8236255110533577,
 'lr': 0.003918784585237195,
 'lr_1': 0.002237071747066137,
 'nb_epoch': 150,
 'pool_size': 2}
best_model,result = keras_fmin_fnct(params_2)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv1d_15 (Conv1D)           (None, 124, 32)           1472      
_________________________________________________________________
conv1d_16 (Conv1D)           (None, 122, 16)           1552      
_________________________________________________________________
dropout_8 (Dropout)          (None, 122, 16)           0         
_________________________________________________________________
max_pooling1d_8 (MaxPooling1 (None, 61, 16)            0         
_________________________________________________________________
flatten_8 (Flatten)          (None, 976)               0         
_________________________________________________________________
dense_15 (Dense)             (None, 64)                62528     
_________________________________________________________________
dense_16 (Dense)             (None, 3)                 195       
=================================================================
Total params: 65,747
Trainable params: 65,747
Non-trainable params: 0
_________________________________________________________________
None
```

```
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-
packages\ipykernel_launcher.py:31: UserWarning: The `nb_epoch` argument in `fit` has been renamed
`epochs`.
```

```
Train on 4067 samples, validate on 1560 samples
Epoch 1/150
4067/4067 [==============================] - 2s 482us/step - loss: 10.6870 - acc: 0.8333 -
val_loss: 3.0266 - val_acc: 0.9006
Epoch 2/150
4067/4067 [==============================] - 1s 325us/step - loss: 1.3128 - acc: 0.8955 -
val_loss: 0.6611 - val_acc: 0.8705
Epoch 3/150
4067/4067 [==============================] - 1s 318us/step - loss: 0.4724 - acc: 0.9036 -
val_loss: 0.5028 - val_acc: 0.8526
Epoch 4/150
4067/4067 [==============================] - 1s 314us/step - loss: 0.3897 - acc: 0.9110 -
val_loss: 0.4572 - val_acc: 0.8538
Epoch 5/150
4067/4067 [==============================] - 1s 321us/step - loss: 0.3501 - acc: 0.9137 -
val_loss: 0.4066 - val_acc: 0.8795
Epoch 6/150
4067/4067 [==============================] - 1s 316us/step - loss: 0.3264 - acc: 0.9166 -
val_loss: 0.3806 - val_acc: 0.8705
Epoch 7/150
4067/4067 [==============================] - 1s 345us/step - loss: 0.3255 - acc: 0.9147 -
val_loss: 0.6140 - val_acc: 0.8571
Epoch 8/150
4067/4067 [==============================] - 1s 344us/step - loss: 0.2770 - acc: 0.9289 -
val_loss: 0.4432 - val_acc: 0.8628
Epoch 9/150
4067/4067 [==============================] - 1s 348us/step - loss: 0.2774 - acc: 0.9275 -
val_loss: 0.2983 - val_acc: 0.9186
Epoch 10/150
4067/4067 [==============================] - 1s 357us/step - loss: 0.2567 - acc: 0.9294 -
val_loss: 0.2868 - val_acc: 0.9071
Epoch 11/150
4067/4067 [==============================] - 2s 384us/step - loss: 0.2484 - acc: 0.9321 -
val_loss: 0.3229 - val_acc: 0.8846
Epoch 12/150
4067/4067 [==============================] - 2s 370us/step - loss: 0.2468 - acc: 0.9331 -
val_loss: 0.2940 - val_acc: 0.9205
Epoch 13/150
```

```
4067/4067 [==============================] - 1s 364us/step - loss: 0.2329 - acc: 0.9375 -
val_loss: 0.3378 - val_acc: 0.8891
Epoch 14/150
4067/4067 [==============================] - 1s 365us/step - loss: 0.2340 - acc: 0.9398 -
val_loss: 0.2612 - val_acc: 0.9308
Epoch 15/150
4067/4067 [==============================] - 1s 361us/step - loss: 0.2185 - acc: 0.9412 -
val_loss: 0.2716 - val_acc: 0.9019
Epoch 16/150
4067/4067 [==============================] - 1s 344us/step - loss: 0.2062 - acc: 0.9425 -
val_loss: 0.3121 - val_acc: 0.8885
Epoch 17/150
4067/4067 [==============================] - 2s 373us/step - loss: 0.1973 - acc: 0.9469 -
val_loss: 0.2888 - val_acc: 0.8987
Epoch 18/150
4067/4067 [==============================] - 1s 354us/step - loss: 0.2661 - acc: 0.9403 -
val_loss: 0.3274 - val_acc: 0.8865
Epoch 19/150
4067/4067 [==============================] - 1s 351us/step - loss: 0.2099 - acc: 0.9412 -
val_loss: 0.2674 - val_acc: 0.9090
Epoch 20/150
4067/4067 [==============================] - 1s 352us/step - loss: 0.1971 - acc: 0.9466 -
val_loss: 0.2634 - val_acc: 0.9058
Epoch 21/150
4067/4067 [==============================] - 1s 349us/step - loss: 0.1965 - acc: 0.9459 -
val_loss: 0.2611 - val_acc: 0.9103
Epoch 22/150
4067/4067 [==============================] - 1s 360us/step - loss: 0.1833 - acc: 0.9506 -
val_loss: 0.2600 - val_acc: 0.9135
Epoch 23/150
4067/4067 [==============================] - 2s 459us/step - loss: 0.2018 - acc: 0.9452 -
val_loss: 0.2637 - val_acc: 0.9058
Epoch 24/150
4067/4067 [==============================] - 1s 340us/step - loss: 0.1962 - acc: 0.9479 -
val_loss: 0.2524 - val_acc: 0.9167
Epoch 25/150
4067/4067 [==============================] - 1s 356us/step - loss: 0.1920 - acc: 0.9469 -
val_loss: 0.2976 - val_acc: 0.8968
Epoch 26/150
4067/4067 [==============================] - 1s 349us/step - loss: 0.1858 - acc: 0.9498 -
val_loss: 0.2361 - val_acc: 0.9410
Epoch 27/150
4067/4067 [==============================] - 1s 343us/step - loss: 0.1799 - acc: 0.9501 -
val_loss: 0.2628 - val_acc: 0.9071
Epoch 28/150
4067/4067 [==============================] - 1s 361us/step - loss: 0.1832 - acc: 0.9560 -
val_loss: 0.2364 - val_acc: 0.9391
Epoch 29/150
4067/4067 [==============================] - 1s 349us/step - loss: 0.1939 - acc: 0.9548 -
val_loss: 0.2562 - val_acc: 0.9314
Epoch 30/150
4067/4067 [==============================] - 1s 348us/step - loss: 0.1749 - acc: 0.9577 -
val_loss: 0.2773 - val_acc: 0.9122
Epoch 31/150
4067/4067 [==============================] - 1s 337us/step - loss: 0.1797 - acc: 0.9503 -
val_loss: 0.2475 - val_acc: 0.9397
Epoch 32/150
4067/4067 [==============================] - 1s 352us/step - loss: 0.1889 - acc: 0.9528 -
val_loss: 0.2490 - val_acc: 0.9160
Epoch 33/150
4067/4067 [==============================] - 2s 373us/step - loss: 0.1719 - acc: 0.9552 -
val_loss: 0.2564 - val_acc: 0.9314
Epoch 34/150
4067/4067 [==============================] - 1s 364us/step - loss: 0.1963 - acc: 0.9521 -
val_loss: 0.2562 - val_acc: 0.9205
Epoch 35/150
4067/4067 [==============================] - 2s 401us/step - loss: 0.1590 - acc: 0.9597 -
val_loss: 0.2717 - val_acc: 0.9378
Epoch 36/150
4067/4067 [==============================] - 1s 357us/step - loss: 0.1807 - acc: 0.9538 -
val_loss: 0.2526 - val_acc: 0.9205
Epoch 37/150
4067/4067 [==============================] - 1s 337us/step - loss: 0.1601 - acc: 0.9587 -
val_loss: 0.2864 - val_acc: 0.9231
Epoch 38/150
4067/4067 [==============================] - 1s 339us/step - loss: 0.1691 - acc: 0.9550 -
val_loss: 0.2342 - val_acc: 0.9321
```

```
Epoch 39/150
4067/4067 [==============================] - 1s 341us/step - loss: 0.1778 - acc: 0.9540 -
val_loss: 0.2449 - val_acc: 0.9474
Epoch 40/150
4067/4067 [==============================] - 1s 348us/step - loss: 0.1795 - acc: 0.9565 -
val_loss: 0.2270 - val_acc: 0.9481
Epoch 41/150
4067/4067 [==============================] - 1s 345us/step - loss: 0.1633 - acc: 0.9592 -
val_loss: 0.3154 - val_acc: 0.9083
Epoch 42/150
4067/4067 [==============================] - 1s 344us/step - loss: 0.1629 - acc: 0.9582 -
val_loss: 0.2632 - val_acc: 0.9410
Epoch 43/150
4067/4067 [==============================] - 1s 341us/step - loss: 0.1707 - acc: 0.9562 -
val_loss: 0.2557 - val_acc: 0.9250
Epoch 44/150
4067/4067 [==============================] - 2s 372us/step - loss: 0.1792 - acc: 0.9577 -
val_loss: 0.2376 - val_acc: 0.9385
Epoch 45/150
4067/4067 [==============================] - 1s 357us/step - loss: 0.1639 - acc: 0.9584 -
val_loss: 0.3097 - val_acc: 0.9006
Epoch 46/150
4067/4067 [==============================] - 1s 349us/step - loss: 0.2030 - acc: 0.9513 -
val_loss: 0.3272 - val_acc: 0.9321
Epoch 47/150
4067/4067 [==============================] - 1s 347us/step - loss: 0.1672 - acc: 0.9562 -
val_loss: 0.2551 - val_acc: 0.9346
Epoch 48/150
4067/4067 [==============================] - 1s 353us/step - loss: 0.1710 - acc: 0.9582 -
val_loss: 0.4064 - val_acc: 0.8795
Epoch 49/150
4067/4067 [==============================] - 1s 344us/step - loss: 0.2007 - acc: 0.9513 -
val_loss: 0.3935 - val_acc: 0.8962
Epoch 50/150
4067/4067 [==============================] - 1s 350us/step - loss: 0.2025 - acc: 0.9538 -
val_loss: 0.3913 - val_acc: 0.8865
Epoch 51/150
4067/4067 [==============================] - 1s 347us/step - loss: 0.1997 - acc: 0.9543 -
val_loss: 0.2883 - val_acc: 0.9224
Epoch 52/150
4067/4067 [==============================] - 1s 360us/step - loss: 0.2003 - acc: 0.9457 -
val_loss: 0.6212 - val_acc: 0.8705
Epoch 53/150
4067/4067 [==============================] - 1s 341us/step - loss: 0.1920 - acc: 0.9525 -
val_loss: 0.2996 - val_acc: 0.9295
Epoch 54/150
4067/4067 [==============================] - 1s 356us/step - loss: 0.2012 - acc: 0.9555 -
val_loss: 0.3581 - val_acc: 0.9045
Epoch 55/150
4067/4067 [==============================] - 1s 345us/step - loss: 0.1985 - acc: 0.9540 -
val_loss: 0.4082 - val_acc: 0.8994
Epoch 56/150
4067/4067 [==============================] - 1s 348us/step - loss: 0.2146 - acc: 0.9508 -
val_loss: 0.3197 - val_acc: 0.9340
Epoch 57/150
4067/4067 [==============================] - 1s 348us/step - loss: 0.2290 - acc: 0.9447 -
val_loss: 0.3018 - val_acc: 0.9449
Epoch 58/150
4067/4067 [==============================] - 1s 352us/step - loss: 0.2379 - acc: 0.9484 -
val_loss: 0.3235 - val_acc: 0.9410
Epoch 59/150
4067/4067 [==============================] - 1s 345us/step - loss: 0.2668 - acc: 0.9449 -
val_loss: 0.3397 - val_acc: 0.9282
Epoch 60/150
4067/4067 [==============================] - 1s 360us/step - loss: 0.2407 - acc: 0.9457 -
val_loss: 0.6286 - val_acc: 0.8846
Epoch 61/150
4067/4067 [==============================] - 1s 340us/step - loss: 0.2392 - acc: 0.9521 -
val_loss: 0.5624 - val_acc: 0.8897
Epoch 62/150
4067/4067 [==============================] - 1s 345us/step - loss: 0.2872 - acc: 0.9378 -
val_loss: 0.3215 - val_acc: 0.9455
Epoch 63/150
4067/4067 [==============================] - 1s 340us/step - loss: 0.2863 - acc: 0.9427 -
val_loss: 0.3915 - val_acc: 0.9256
Epoch 64/150
4067/4067 [==============================] - 1s 341us/step - loss: 0.3255 - acc: 0.9430 -
```

```
val_loss: 0.6421 - val_acc: 0.8974
Epoch 65/150
4067/4067 [==============================] - 2s 433us/step - loss: 0.3042 - acc: 0.9420 -
val_loss: 1.1776 - val_acc: 0.8333
Epoch 66/150
4067/4067 [==============================] - 1s 357us/step - loss: 0.3357 - acc: 0.9417 -
val_loss: 1.2749 - val_acc: 0.8340
Epoch 67/150
4067/4067 [==============================] - 1s 352us/step - loss: 0.3750 - acc: 0.9412 -
val_loss: 0.5633 - val_acc: 0.9167
Epoch 68/150
4067/4067 [==============================] - 1s 353us/step - loss: 0.3569 - acc: 0.9412 -
val_loss: 0.3629 - val_acc: 0.9397
Epoch 69/150
4067/4067 [==============================] - 1s 341us/step - loss: 0.3613 - acc: 0.9484 -
val_loss: 0.8218 - val_acc: 0.8795
Epoch 70/150
4067/4067 [==============================] - 1s 356us/step - loss: 0.4420 - acc: 0.9363 -
val_loss: 0.6252 - val_acc: 0.9205
Epoch 71/150
4067/4067 [==============================] - 1s 355us/step - loss: 0.3416 - acc: 0.9538 -
val_loss: 0.8496 - val_acc: 0.8846
Epoch 72/150
4067/4067 [==============================] - 1s 340us/step - loss: 0.4581 - acc: 0.9373 -
val_loss: 0.5673 - val_acc: 0.9378
Epoch 73/150
4067/4067 [==============================] - 1s 342us/step - loss: 0.4212 - acc: 0.9538 -
val_loss: 0.4301 - val_acc: 0.9429
Epoch 74/150
4067/4067 [==============================] - 1s 344us/step - loss: 0.5009 - acc: 0.9393 -
val_loss: 0.6020 - val_acc: 0.9179
Epoch 75/150
4067/4067 [==============================] - 1s 352us/step - loss: 0.4760 - acc: 0.9444 -
val_loss: 0.5703 - val_acc: 0.9346
Epoch 76/150
4067/4067 [==============================] - 1s 341us/step - loss: 0.4615 - acc: 0.9454 -
val_loss: 1.8877 - val_acc: 0.8288
Epoch 77/150
4067/4067 [==============================] - 1s 355us/step - loss: 0.5637 - acc: 0.9400 -
val_loss: 0.7263 - val_acc: 0.9115
Epoch 78/150
4067/4067 [==============================] - 1s 356us/step - loss: 0.4969 - acc: 0.9486 -
val_loss: 1.4533 - val_acc: 0.8596
Epoch 79/150
4067/4067 [==============================] - ETA: 0s - loss: 0.5597 - acc: 0.9415- ETA: 0s - loss:
0.5276 - 1s 353us/step - loss: 0.5594 - acc: 0.9415 - val_loss: 1.3423 - val_acc: 0.8833
Epoch 80/150
4067/4067 [==============================] - 1s 352us/step - loss: 0.6181 - acc: 0.9459 -
val_loss: 0.6729 - val_acc: 0.9372
Epoch 81/150
4067/4067 [==============================] - 1s 349us/step - loss: 0.5254 - acc: 0.9466 -
val_loss: 0.6592 - val_acc: 0.9353
Epoch 82/150
4067/4067 [==============================] - 1s 360us/step - loss: 0.6281 - acc: 0.9427 -
val_loss: 0.6276 - val_acc: 0.9365
Epoch 83/150
4067/4067 [==============================] - 1s 344us/step - loss: 0.6015 - acc: 0.9437 -
val_loss: 0.6181 - val_acc: 0.9506
Epoch 84/150
4067/4067 [==============================] - 1s 349us/step - loss: 0.6184 - acc: 0.9471 -
val_loss: 1.0193 - val_acc: 0.9000
Epoch 85/150
4067/4067 [==============================] - 1s 352us/step - loss: 0.6341 - acc: 0.9439 -
val_loss: 0.6097 - val_acc: 0.9538
Epoch 86/150
4067/4067 [==============================] - 1s 352us/step - loss: 0.6224 - acc: 0.9444 -
val_loss: 1.0916 - val_acc: 0.9199
Epoch 87/150
4067/4067 [==============================] - 1s 351us/step - loss: 0.6279 - acc: 0.9484 -
val_loss: 1.0537 - val_acc: 0.9103
Epoch 88/150
4067/4067 [==============================] - 1s 342us/step - loss: 0.7142 - acc: 0.9439 -
val_loss: 0.8499 - val_acc: 0.9372
Epoch 89/150
4067/4067 [==============================] - 1s 348us/step - loss: 0.7863 - acc: 0.9432 -
val_loss: 0.9607 - val_acc: 0.9218
Epoch 90/150
```

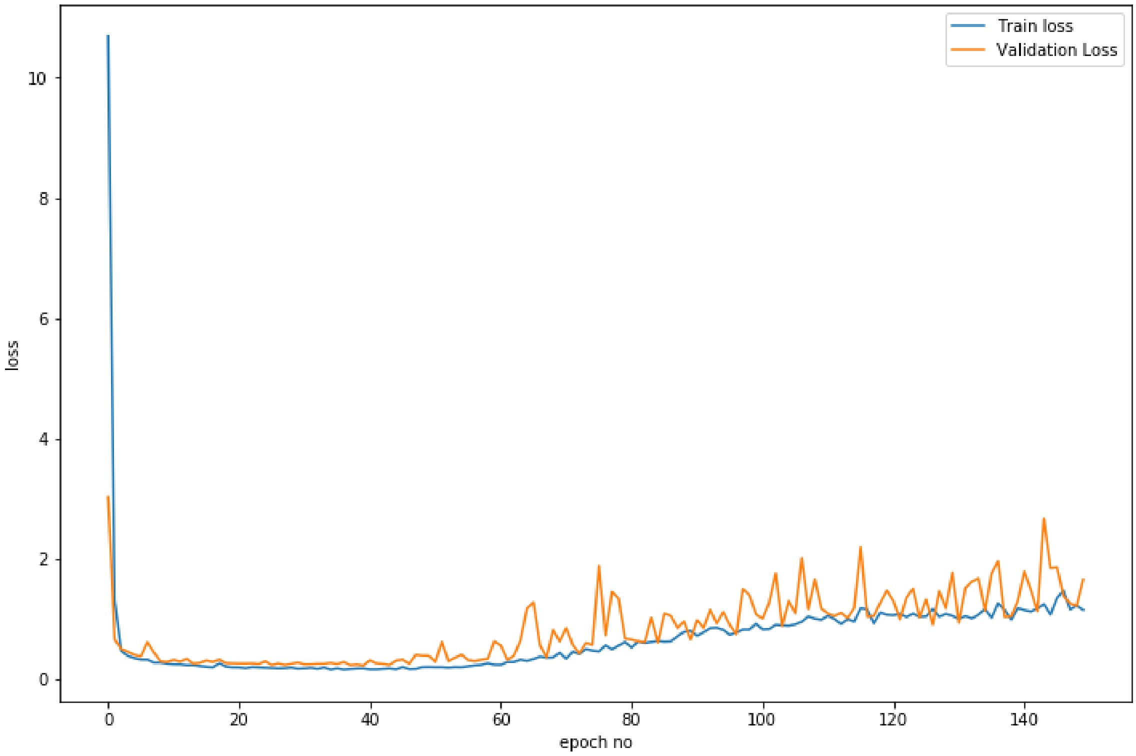
```
4067/4067 [==============================] - 2s 371us/step - loss: 0.8041 - acc: 0.9462 - val_loss: 0.6598 - val_acc: 0.9519
Epoch 91/150
4067/4067 [==============================] - 2s 442us/step - loss: 0.7227 - acc: 0.9508 - val_loss: 0.9813 - val_acc: 0.9256
Epoch 92/150
4067/4067 [==============================] - 2s 416us/step - loss: 0.7791 - acc: 0.9479 - val_loss: 0.8503 - val_acc: 0.9365
Epoch 93/150
4067/4067 [==============================] - 2s 430us/step - loss: 0.8488 - acc: 0.9422 - val_loss: 1.1558 - val_acc: 0.9212
Epoch 94/150
4067/4067 [==============================] - 2s 495us/step - loss: 0.8559 - acc: 0.9444 - val_loss: 0.9262 - val_acc: 0.9378
Epoch 95/150
4067/4067 [==============================] - 2s 394us/step - loss: 0.8195 - acc: 0.9484 - val_loss: 1.1169 - val_acc: 0.9263
Epoch 96/150
4067/4067 [==============================] - 1s 360us/step - loss: 0.7387 - acc: 0.9545 - val_loss: 0.9006 - val_acc: 0.9385
Epoch 97/150
4067/4067 [==============================] - 2s 372us/step - loss: 0.7764 - acc: 0.9508 - val_loss: 0.7460 - val_acc: 0.9513
Epoch 98/150
4067/4067 [==============================] - 1s 353us/step - loss: 0.8253 - acc: 0.9489 - val_loss: 1.4987 - val_acc: 0.8872
Epoch 99/150
4067/4067 [==============================] - 1s 344us/step - loss: 0.8257 - acc: 0.9503 - val_loss: 1.3966 - val_acc: 0.9115
Epoch 100/150
4067/4067 [==============================] - 1s 365us/step - loss: 0.9226 - acc: 0.9405 - val_loss: 1.0780 - val_acc: 0.9327
Epoch 101/150
4067/4067 [==============================] - 1s 352us/step - loss: 0.8283 - acc: 0.9516 - val_loss: 1.0076 - val_acc: 0.9385
Epoch 102/150
4067/4067 [==============================] - 1s 365us/step - loss: 0.8333 - acc: 0.9503 - val_loss: 1.2687 - val_acc: 0.9147
Epoch 103/150
4067/4067 [==============================] - 1s 344us/step - loss: 0.9023 - acc: 0.9486 - val_loss: 1.7615 - val_acc: 0.8859
Epoch 104/150
4067/4067 [==============================] - 1s 344us/step - loss: 0.8954 - acc: 0.9469 - val_loss: 0.8835 - val_acc: 0.9468
Epoch 105/150
4067/4067 [==============================] - 1s 348us/step - loss: 0.8882 - acc: 0.9501 - val_loss: 1.3025 - val_acc: 0.9205
Epoch 106/150
4067/4067 [==============================] - 2s 410us/step - loss: 0.9107 - acc: 0.9464 - val_loss: 1.0963 - val_acc: 0.9346
Epoch 107/150
4067/4067 [==============================] - 2s 394us/step - loss: 0.9556 - acc: 0.9452 - val_loss: 2.0161 - val_acc: 0.8795
Epoch 108/150
4067/4067 [==============================] - 1s 353us/step - loss: 1.0421 - acc: 0.9400 - val_loss: 1.1681 - val_acc: 0.9301
Epoch 109/150
4067/4067 [==============================] - 1s 352us/step - loss: 1.0061 - acc: 0.9457 - val_loss: 1.6619 - val_acc: 0.8974
Epoch 110/150
4067/4067 [==============================] - 1s 357us/step - loss: 0.9856 - acc: 0.9459 - val_loss: 1.1742 - val_acc: 0.9308
Epoch 111/150
4067/4067 [==============================] - 1s 348us/step - loss: 1.0574 - acc: 0.9425 - val_loss: 1.0911 - val_acc: 0.9385
Epoch 112/150
4067/4067 [==============================] - 1s 344us/step - loss: 0.9989 - acc: 0.9444 - val_loss: 1.0512 - val_acc: 0.9397
Epoch 113/150
4067/4067 [==============================] - 1s 360us/step - loss: 0.9164 - acc: 0.9489 - val_loss: 1.1036 - val_acc: 0.9359
Epoch 114/150
4067/4067 [==============================] - 1s 351us/step - loss: 0.9962 - acc: 0.9466 - val_loss: 1.0179 - val_acc: 0.9455
Epoch 115/150
4067/4067 [==============================] - 1s 354us/step - loss: 0.9582 - acc: 0.9496 - val_loss: 1.1834 - val_acc: 0.9359
```

```
Epoch 116/150
4067/4067 [==============================] - 1s 345us/step - loss: 1.1802 - acc: 0.9351 -
val_loss: 2.1964 - val_acc: 0.8731
Epoch 117/150
4067/4067 [==============================] - 1s 348us/step - loss: 1.1621 - acc: 0.9358 -
val_loss: 1.0396 - val_acc: 0.9436
Epoch 118/150
4067/4067 [==============================] - 1s 340us/step - loss: 0.9269 - acc: 0.9523 -
val_loss: 1.0535 - val_acc: 0.9385
Epoch 119/150
4067/4067 [==============================] - 1s 344us/step - loss: 1.1087 - acc: 0.9375 -
val_loss: 1.2717 - val_acc: 0.9308
Epoch 120/150
4067/4067 [==============================] - 1s 344us/step - loss: 1.0718 - acc: 0.9427 -
val_loss: 1.4719 - val_acc: 0.9147
Epoch 121/150
4067/4067 [==============================] - 1s 332us/step - loss: 1.0634 - acc: 0.9430 -
val_loss: 1.2992 - val_acc: 0.9250
Epoch 122/150
4067/4067 [==============================] - 1s 336us/step - loss: 1.0872 - acc: 0.9430 -
val_loss: 0.9939 - val_acc: 0.9462
Epoch 123/150
4067/4067 [==============================] - 1s 336us/step - loss: 1.0327 - acc: 0.9444 -
val_loss: 1.3579 - val_acc: 0.9205 0s - loss: 1.1716
Epoch 124/150
4067/4067 [==============================] - 1s 332us/step - loss: 1.0902 - acc: 0.9393 -
val_loss: 1.5052 - val_acc: 0.9147
Epoch 125/150
4067/4067 [==============================] - 1s 336us/step - loss: 1.0375 - acc: 0.9442 -
val_loss: 1.0277 - val_acc: 0.9429
Epoch 126/150
4067/4067 [==============================] - 1s 328us/step - loss: 1.0444 - acc: 0.9439 -
val_loss: 1.3256 - val_acc: 0.9218
Epoch 127/150
4067/4067 [==============================] - 1s 340us/step - loss: 1.1681 - acc: 0.9358 -
val_loss: 0.9073 - val_acc: 0.9494
Epoch 128/150
4067/4067 [==============================] - 1s 332us/step - loss: 1.0381 - acc: 0.9425 -
val_loss: 1.4652 - val_acc: 0.9141
Epoch 129/150
4067/4067 [==============================] - 1s 332us/step - loss: 1.0867 - acc: 0.9400 -
val_loss: 1.1768 - val_acc: 0.9327
Epoch 130/150
4067/4067 [==============================] - 2s 397us/step - loss: 1.0532 - acc: 0.9449 -
val_loss: 1.7736 - val_acc: 0.8974
Epoch 131/150
4067/4067 [==============================] - 2s 426us/step - loss: 1.0052 - acc: 0.9466 -
val_loss: 0.9367 - val_acc: 0.9519
Epoch 132/150
4067/4067 [==============================] - 1s 340us/step - loss: 1.0478 - acc: 0.9434 -
val_loss: 1.5190 - val_acc: 0.9090
Epoch 133/150
4067/4067 [==============================] - 1s 336us/step - loss: 1.0107 - acc: 0.9449 -
val_loss: 1.6232 - val_acc: 0.9058
Epoch 134/150
4067/4067 [==============================] - 1s 332us/step - loss: 1.0720 - acc: 0.9422 -
val_loss: 1.6769 - val_acc: 0.9051
Epoch 135/150
4067/4067 [==============================] - 1s 343us/step - loss: 1.1717 - acc: 0.9373 -
val_loss: 1.1355 - val_acc: 0.9359
Epoch 136/150
4067/4067 [==============================] - 1s 345us/step - loss: 1.0188 - acc: 0.9462 -
val_loss: 1.7602 - val_acc: 0.8962
Epoch 137/150
4067/4067 [==============================] - 1s 340us/step - loss: 1.2609 - acc: 0.9307 -
val_loss: 1.9604 - val_acc: 0.8878
Epoch 138/150
4067/4067 [==============================] - 1s 331us/step - loss: 1.1513 - acc: 0.9375 -
val_loss: 1.0344 - val_acc: 0.9468
Epoch 139/150
4067/4067 [==============================] - 1s 329us/step - loss: 0.9899 - acc: 0.9491 -
val_loss: 1.0471 - val_acc: 0.9410
Epoch 140/150
4067/4067 [==============================] - 1s 336us/step - loss: 1.1802 - acc: 0.9373 -
val_loss: 1.2992 - val_acc: 0.9288
Epoch 141/150
4067/4067 [==============================] - 1s 340us/step - loss: 1.1490 - acc: 0.9380 -
```

```
val_loss: 1.7921 - val_acc: 0.8994
Epoch 142/150
4067/4067 [==============================] - 1s 353us/step - loss: 1.1242 - acc: 0.9395 -
val_loss: 1.4830 - val_acc: 0.9167
Epoch 143/150
4067/4067 [==============================] - 1s 344us/step - loss: 1.1758 - acc: 0.9366 -
val_loss: 1.1290 - val_acc: 0.9404
Epoch 144/150
4067/4067 [==============================] - 1s 336us/step - loss: 1.2460 - acc: 0.9331 -
val_loss: 2.6686 - val_acc: 0.8436
Epoch 145/150
4067/4067 [==============================] - 1s 339us/step - loss: 1.0744 - acc: 0.9434 -
val_loss: 1.8467 - val_acc: 0.8955
Epoch 146/150
4067/4067 [==============================] - 1s 328us/step - loss: 1.3486 - acc: 0.9272 -
val_loss: 1.8590 - val_acc: 0.8904
Epoch 147/150
4067/4067 [==============================] - 1s 336us/step - loss: 1.4748 - acc: 0.9191 -
val_loss: 1.3827 - val_acc: 0.9237
Epoch 148/150
4067/4067 [==============================] - 1s 340us/step - loss: 1.1593 - acc: 0.9375 -
val_loss: 1.2578 - val_acc: 0.9288
Epoch 149/150
4067/4067 [==============================] - 1s 361us/step - loss: 1.2212 - acc: 0.9331 -
val_loss: 1.2208 - val_acc: 0.9301
Epoch 150/150
4067/4067 [==============================] - 2s 420us/step - loss: 1.1523 - acc: 0.9383 -
val_loss: 1.6555 - val_acc: 0.9077
```

In [89]:

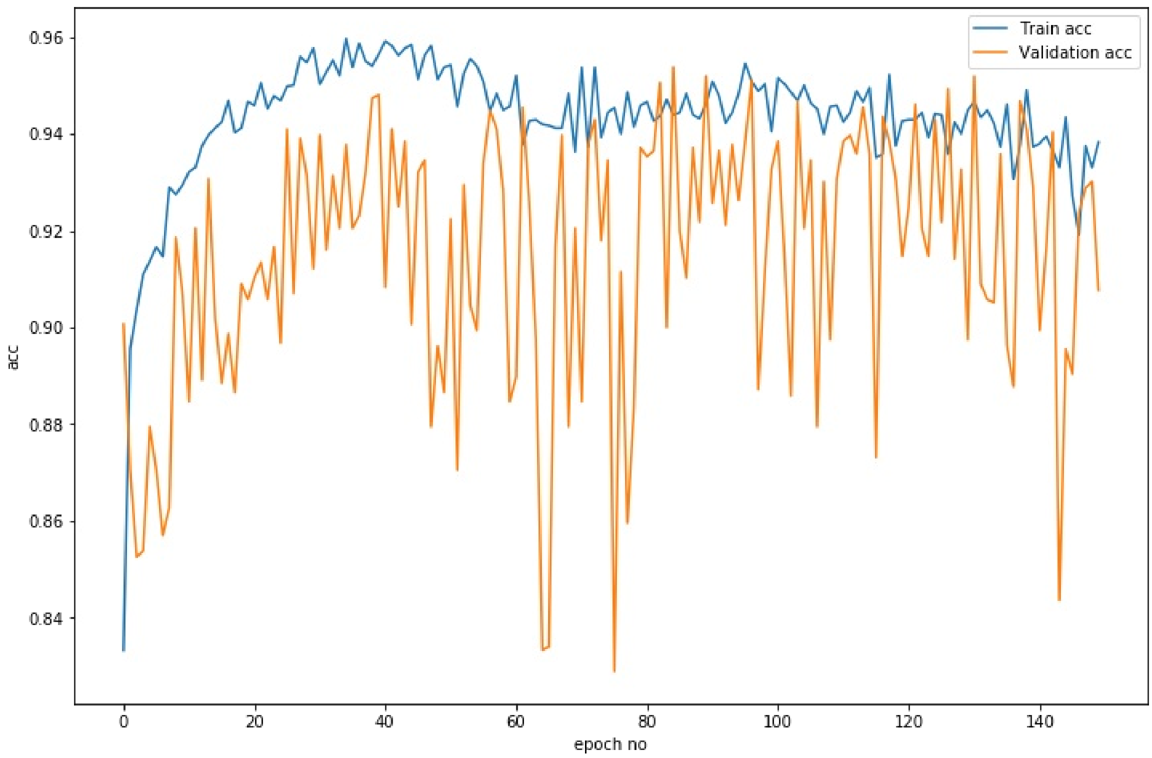
```
plt.figure(figsize=(12,8))
plt.plot(result.history['loss'],label='Train loss')
plt.plot(result.history['val_loss'],label = 'Validation Loss')
plt.xlabel('epoch no')
plt.ylabel('loss')
plt.legend()
plt.show()
```




In [90]:

```
plt.figure(figsize=(12,8))
plt.plot(result.history['acc'],label='Train acc')
plt.plot(result.history['val_acc'],label = 'Validation acc')
plt.xlabel('epoch no')
```

```
plt.ylabel('acc')
plt.legend()
plt.show()
```




around 83-85 score is giving good accuracy wit less overfitting

In [91]:

```
params_3={'Dense': 64,
 'Dense_1': 64,
 'Dropout': 0.45377377480700615,
 'choiceval': 'rmsprop',
 'filters': 32,
 'filters_1': 16,
 'kernel_size': 5,
 'kernel_size_1': 3,
 'l2': 0.0019801221163149862,
 'l2_1': 0.8236255110533577,
 'lr': 0.003918784585237195,
 'lr_1': 0.002237071747066137,
 'nb_epoch': 85,
 'pool_size': 2}
best_model,result = keras_fmin_fnct(params_3)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv1d_17 (Conv1D)           (None, 124, 32)           1472      
_________________________________________________________________
conv1d_18 (Conv1D)           (None, 122, 16)           1552      
_________________________________________________________________
dropout_9 (Dropout)          (None, 122, 16)           0         
_________________________________________________________________
max_pooling1d_9 (MaxPooling1 (None, 61, 16)            0         
_________________________________________________________________
flatten_9 (Flatten)          (None, 976)               0         
_________________________________________________________________
dense_17 (Dense)             (None, 64)                62528     
_________________________________________________________________
dense_18 (Dense)             (None, 3)                 195       
=================================================================
Total params: 65,747
Trainable params: 65,747
Non-trainable params: 0
```

```
_________________________________________________________________
None
```

```
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-packages\ipykernel_launcher.py:31: UserWarning: The `nb_epoch` argument in `fit` has been renamed `epochs`.
```

```
Train on 4067 samples, validate on 1560 samples
Epoch 1/85
4067/4067 [==============================] - 2s 514us/step - loss: 10.6870 - acc: 0.8333 - val_loss: 3.0266 - val_acc: 0.9006
Epoch 2/85
4067/4067 [==============================] - 1s 365us/step - loss: 1.3128 - acc: 0.8955 - val_loss: 0.6611 - val_acc: 0.8705
Epoch 3/85
4067/4067 [==============================] - 1s 357us/step - loss: 0.4724 - acc: 0.9036 - val_loss: 0.5028 - val_acc: 0.8526
Epoch 4/85
4067/4067 [==============================] - 1s 334us/step - loss: 0.3897 - acc: 0.9110 - val_loss: 0.4572 - val_acc: 0.8538
Epoch 5/85
4067/4067 [==============================] - 1s 345us/step - loss: 0.3501 - acc: 0.9137 - val_loss: 0.4066 - val_acc: 0.8795
Epoch 6/85
4067/4067 [==============================] - 1s 336us/step - loss: 0.3264 - acc: 0.9166 - val_loss: 0.3806 - val_acc: 0.8705
Epoch 7/85
4067/4067 [==============================] - 1s 328us/step - loss: 0.3255 - acc: 0.9147 - val_loss: 0.6140 - val_acc: 0.8571
Epoch 8/85
4067/4067 [==============================] - 1s 331us/step - loss: 0.2770 - acc: 0.9289 - val_loss: 0.4432 - val_acc: 0.8628
Epoch 9/85
4067/4067 [==============================] - 1s 340us/step - loss: 0.2774 - acc: 0.9275 - val_loss: 0.2983 - val_acc: 0.9186
Epoch 10/85
4067/4067 [==============================] - 1s 324us/step - loss: 0.2567 - acc: 0.9294 - val_loss: 0.2868 - val_acc: 0.9071
Epoch 11/85
4067/4067 [==============================] - 1s 332us/step - loss: 0.2484 - acc: 0.9321 - val_loss: 0.3229 - val_acc: 0.8846
Epoch 12/85
4067/4067 [==============================] - 1s 336us/step - loss: 0.2468 - acc: 0.9331 - val_loss: 0.2940 - val_acc: 0.9205
Epoch 13/85
4067/4067 [==============================] - 1s 349us/step - loss: 0.2329 - acc: 0.9375 - val_loss: 0.3378 - val_acc: 0.8891
Epoch 14/85
4067/4067 [==============================] - 2s 408us/step - loss: 0.2340 - acc: 0.9398 - val_loss: 0.2612 - val_acc: 0.9308
Epoch 15/85
4067/4067 [==============================] - 2s 397us/step - loss: 0.2185 - acc: 0.9412 - val_loss: 0.2716 - val_acc: 0.9019
Epoch 16/85
4067/4067 [==============================] - 2s 381us/step - loss: 0.2062 - acc: 0.9425 - val_loss: 0.3121 - val_acc: 0.8885
Epoch 17/85
4067/4067 [==============================] - 2s 410us/step - loss: 0.1973 - acc: 0.9469 - val_loss: 0.2888 - val_acc: 0.8987
Epoch 18/85
4067/4067 [==============================] - 1s 364us/step - loss: 0.2661 - acc: 0.9403 - val_loss: 0.3274 - val_acc: 0.8865
Epoch 19/85
4067/4067 [==============================] - 1s 356us/step - loss: 0.2099 - acc: 0.9412 - val_loss: 0.2674 - val_acc: 0.9090
Epoch 20/85
4067/4067 [==============================] - 1s 344us/step - loss: 0.1971 - acc: 0.9466 - val_loss: 0.2634 - val_acc: 0.9058
Epoch 21/85
4067/4067 [==============================] - 1s 344us/step - loss: 0.1965 - acc: 0.9459 - val_loss: 0.2611 - val_acc: 0.9103
Epoch 22/85
4067/4067 [==============================] - 1s 352us/step - loss: 0.1833 - acc: 0.9506 - val_loss: 0.2600 - val_acc: 0.9135
Epoch 23/85
4067/4067 [==============================] - 1s 345us/step - loss: 0.2018 - acc: 0.9452 -
```

```
val_loss: 0.2637 - val_acc: 0.9058
Epoch 24/85
4067/4067 [==============================] - 1s 351us/step - loss: 0.1962 - acc: 0.9479 -
val_loss: 0.2524 - val_acc: 0.9167
Epoch 25/85
4067/4067 [==============================] - 1s 357us/step - loss: 0.1920 - acc: 0.9469 -
val_loss: 0.2976 - val_acc: 0.8968
Epoch 26/85
4067/4067 [==============================] - 1s 354us/step - loss: 0.1858 - acc: 0.9498 -
val_loss: 0.2361 - val_acc: 0.9410
Epoch 27/85
4067/4067 [==============================] - 1s 355us/step - loss: 0.1799 - acc: 0.9501 -
val_loss: 0.2628 - val_acc: 0.9071
Epoch 28/85
4067/4067 [==============================] - 1s 349us/step - loss: 0.1832 - acc: 0.9560 -
val_loss: 0.2364 - val_acc: 0.9391
Epoch 29/85
4067/4067 [==============================] - 1s 348us/step - loss: 0.1939 - acc: 0.9548 -
val_loss: 0.2562 - val_acc: 0.9314
Epoch 30/85
4067/4067 [==============================] - 1s 353us/step - loss: 0.1749 - acc: 0.9577 -
val_loss: 0.2773 - val_acc: 0.9122
Epoch 31/85
4067/4067 [==============================] - 1s 352us/step - loss: 0.1797 - acc: 0.9503 -
val_loss: 0.2475 - val_acc: 0.9397
Epoch 32/85
4067/4067 [==============================] - 1s 352us/step - loss: 0.1889 - acc: 0.9528 -
val_loss: 0.2490 - val_acc: 0.9160
Epoch 33/85
4067/4067 [==============================] - 1s 356us/step - loss: 0.1719 - acc: 0.9552 -
val_loss: 0.2564 - val_acc: 0.9314
Epoch 34/85
4067/4067 [==============================] - 1s 361us/step - loss: 0.1963 - acc: 0.9521 -
val_loss: 0.2562 - val_acc: 0.9205
Epoch 35/85
4067/4067 [==============================] - 1s 353us/step - loss: 0.1590 - acc: 0.9597 -
val_loss: 0.2717 - val_acc: 0.9378
Epoch 36/85
4067/4067 [==============================] - 1s 360us/step - loss: 0.1807 - acc: 0.9538 -
val_loss: 0.2526 - val_acc: 0.9205
Epoch 37/85
4067/4067 [==============================] - 1s 353us/step - loss: 0.1601 - acc: 0.9587 -
val_loss: 0.2864 - val_acc: 0.9231
Epoch 38/85
4067/4067 [==============================] - 1s 356us/step - loss: 0.1691 - acc: 0.9550 -
val_loss: 0.2342 - val_acc: 0.9321
Epoch 39/85
4067/4067 [==============================] - 1s 365us/step - loss: 0.1778 - acc: 0.9540 -
val_loss: 0.2449 - val_acc: 0.9474
Epoch 40/85
4067/4067 [==============================] - 1s 363us/step - loss: 0.1795 - acc: 0.9565 -
val_loss: 0.2270 - val_acc: 0.9481
Epoch 41/85
4067/4067 [==============================] - 2s 378us/step - loss: 0.1633 - acc: 0.9592 -
val_loss: 0.3154 - val_acc: 0.9083
Epoch 42/85
4067/4067 [==============================] - 2s 390us/step - loss: 0.1629 - acc: 0.9582 -
val_loss: 0.2632 - val_acc: 0.9410
Epoch 43/85
4067/4067 [==============================] - 1s 369us/step - loss: 0.1707 - acc: 0.9562 -
val_loss: 0.2557 - val_acc: 0.9250
Epoch 44/85
4067/4067 [==============================] - 1s 351us/step - loss: 0.1792 - acc: 0.9577 -
val_loss: 0.2376 - val_acc: 0.9385
Epoch 45/85
4067/4067 [==============================] - 2s 375us/step - loss: 0.1639 - acc: 0.9584 -
val_loss: 0.3097 - val_acc: 0.9006
Epoch 46/85
4067/4067 [==============================] - 2s 374us/step - loss: 0.2030 - acc: 0.9513 -
val_loss: 0.3272 - val_acc: 0.9321
Epoch 47/85
4067/4067 [==============================] - 1s 367us/step - loss: 0.1672 - acc: 0.9562 -
val_loss: 0.2551 - val_acc: 0.9346
Epoch 48/85
4067/4067 [==============================] - 1s 357us/step - loss: 0.1710 - acc: 0.9582 -
val_loss: 0.4064 - val_acc: 0.8795
Epoch 49/85
```

```
4067/4067 [==============================] - ETA: 0s - loss: 0.2003 - acc: 0.951 - 2s 369us/step -
loss: 0.2007 - acc: 0.9513 - val_loss: 0.3935 - val_acc: 0.8962
Epoch 50/85
4067/4067 [==============================] - 1s 360us/step - loss: 0.2025 - acc: 0.9538 -
val_loss: 0.3913 - val_acc: 0.8865
Epoch 51/85
4067/4067 [==============================] - 1s 356us/step - loss: 0.1997 - acc: 0.9543 -
val_loss: 0.2883 - val_acc: 0.9224
Epoch 52/85
4067/4067 [==============================] - 1s 364us/step - loss: 0.2003 - acc: 0.9457 -
val_loss: 0.6212 - val_acc: 0.8705
Epoch 53/85
4067/4067 [==============================] - 1s 356us/step - loss: 0.1920 - acc: 0.9525 -
val_loss: 0.2996 - val_acc: 0.9295
Epoch 54/85
4067/4067 [==============================] - 1s 357us/step - loss: 0.2012 - acc: 0.9555 -
val_loss: 0.3581 - val_acc: 0.9045
Epoch 55/85
4067/4067 [==============================] - 2s 401us/step - loss: 0.1985 - acc: 0.9540 -
val_loss: 0.4082 - val_acc: 0.8994
Epoch 56/85
4067/4067 [==============================] - 2s 426us/step - loss: 0.2146 - acc: 0.9508 -
val_loss: 0.3197 - val_acc: 0.9340
Epoch 57/85
4067/4067 [==============================] - 1s 356us/step - loss: 0.2290 - acc: 0.9447 -
val_loss: 0.3018 - val_acc: 0.9449
Epoch 58/85
4067/4067 [==============================] - 1s 357us/step - loss: 0.2379 - acc: 0.9484 -
val_loss: 0.3235 - val_acc: 0.9410
Epoch 59/85
4067/4067 [==============================] - 1s 357us/step - loss: 0.2668 - acc: 0.9449 -
val_loss: 0.3397 - val_acc: 0.9282
Epoch 60/85
4067/4067 [==============================] - 1s 352us/step - loss: 0.2407 - acc: 0.9457 -
val_loss: 0.6286 - val_acc: 0.8846
Epoch 61/85
4067/4067 [==============================] - 1s 365us/step - loss: 0.2392 - acc: 0.9521 -
val_loss: 0.5624 - val_acc: 0.8897
Epoch 62/85
4067/4067 [==============================] - 2s 377us/step - loss: 0.2872 - acc: 0.9378 -
val_loss: 0.3215 - val_acc: 0.9455
Epoch 63/85
4067/4067 [==============================] - 1s 337us/step - loss: 0.2863 - acc: 0.9427 -
val_loss: 0.3915 - val_acc: 0.9256
Epoch 64/85
4067/4067 [==============================] - 1s 352us/step - loss: 0.3255 - acc: 0.9430 -
val_loss: 0.6421 - val_acc: 0.8974
Epoch 65/85
4067/4067 [==============================] - 1s 347us/step - loss: 0.3042 - acc: 0.9420 -
val_loss: 1.1776 - val_acc: 0.8333
Epoch 66/85
4067/4067 [==============================] - 1s 351us/step - loss: 0.3357 - acc: 0.9417 -
val_loss: 1.2749 - val_acc: 0.8340
Epoch 67/85
4067/4067 [==============================] - 1s 351us/step - loss: 0.3750 - acc: 0.9412 -
val_loss: 0.5633 - val_acc: 0.9167
Epoch 68/85
4067/4067 [==============================] - 1s 356us/step - loss: 0.3569 - acc: 0.9412 -
val_loss: 0.3629 - val_acc: 0.9397
Epoch 69/85
4067/4067 [==============================] - 1s 360us/step - loss: 0.3613 - acc: 0.9484 -
val_loss: 0.8218 - val_acc: 0.8795
Epoch 70/85
4067/4067 [==============================] - 1s 365us/step - loss: 0.4420 - acc: 0.9363 -
val_loss: 0.6252 - val_acc: 0.9205
Epoch 71/85
4067/4067 [==============================] - ETA: 0s - loss: 0.3467 - acc: 0.952 - 1s 357us/step -
loss: 0.3416 - acc: 0.9538 - val_loss: 0.8496 - val_acc: 0.8846
Epoch 72/85
4067/4067 [==============================] - 1s 356us/step - loss: 0.4581 - acc: 0.9373 -
val_loss: 0.5673 - val_acc: 0.9378
Epoch 73/85
4067/4067 [==============================] - 2s 369us/step - loss: 0.4212 - acc: 0.9538 -
val_loss: 0.4301 - val_acc: 0.9429
Epoch 74/85
4067/4067 [==============================] - 1s 353us/step - loss: 0.5009 - acc: 0.9393 -
val_loss: 0.6020 - val_acc: 0.9179
```

```
Epoch 75/85
4067/4067 [==============================] - 1s 345us/step - loss: 0.4760 - acc: 0.9444 -
val_loss: 0.5703 - val_acc: 0.9346
Epoch 76/85
4067/4067 [==============================] - 1s 357us/step - loss: 0.4615 - acc: 0.9454 -
val_loss: 1.8877 - val_acc: 0.8288
Epoch 77/85
4067/4067 [==============================] - 1s 357us/step - loss: 0.5637 - acc: 0.9400 -
val_loss: 0.7263 - val_acc: 0.9115
Epoch 78/85
4067/4067 [==============================] - 2s 448us/step - loss: 0.4969 - acc: 0.9486 -
val_loss: 1.4533 - val_acc: 0.8596
Epoch 79/85
4067/4067 [==============================] - 2s 428us/step - loss: 0.5594 - acc: 0.9415 -
val_loss: 1.3423 - val_acc: 0.8833
Epoch 80/85
4067/4067 [==============================] - 2s 437us/step - loss: 0.6181 - acc: 0.9459 -
val_loss: 0.6729 - val_acc: 0.9372
Epoch 81/85
4067/4067 [==============================] - 2s 434us/step - loss: 0.5254 - acc: 0.9466 -
val_loss: 0.6592 - val_acc: 0.9353
Epoch 82/85
4067/4067 [==============================] - 2s 418us/step - loss: 0.6281 - acc: 0.9427 -
val_loss: 0.6276 - val_acc: 0.9365
Epoch 83/85
4067/4067 [==============================] - 2s 376us/step - loss: 0.6015 - acc: 0.9437 -
val_loss: 0.6181 - val_acc: 0.9506
Epoch 84/85
4067/4067 [==============================] - 2s 382us/step - loss: 0.6184 - acc: 0.9471 -
val_loss: 1.0193 - val_acc: 0.9000
Epoch 85/85
4067/4067 [==============================] - 1s 364us/step - loss: 0.6341 - acc: 0.9439 -
val_loss: 0.6097 - val_acc: 0.9538
```

In [92]:

```
_,acc_test = best_model.evaluate(X_test_s,Y_test_s,verbose=0)
_,acc_train = best_model.evaluate(X_train_s,Y_train_s,verbose=0)
print('Train_accuracy',acc_train,'test_accuracy',acc_test)
```

```
Train_accuracy 0.960904843865257 test_accuracy 0.9538461538461539
```

In [93]:

```
# Confusion Matrix
# Activities are the class labels
# It is a 3 class classification
from sklearn import metrics
ACTIVITIES = {
    0: 'SITTING',
    1: 'STANDING',
    2: 'LAYING',
}
# Utility function to print the confusion matrix
def confusion_matrix_cnn(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'])
    return metrics.confusion_matrix(Y_true, Y_pred)

# Confusion Matrix
print(confusion_matrix_cnn(Y_test_s, best_model.predict(X_test_s)))
```

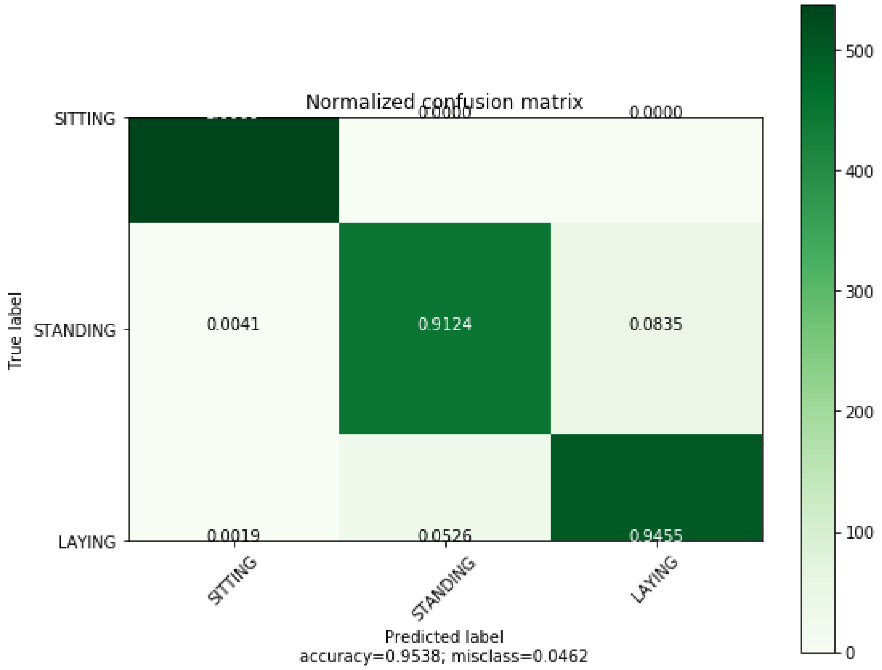
```
[[537   0   0]
 [  2 448  41]
 [  1  28 503]]
```

In [94]:

```
plt.figure(figsize=(8,8))
cm = confusion_matrix_cnn(Y_test_s, best_model.predict(X_test_s))
plot_confusion_matrix(cm, target_names=['SITTING','STANDING','LAYING'], normalize=True,
```

```
title='Normalized confusion matrix', cmap = plt.cm.Greens)
plt.show()
```

```
<Figure size 576x576 with 0 Axes>
```




In [95]:

```
best_model.save('final_model_static.h5')
```

## Classification of Dynamic activities

In [96]:

```
##data preparation
def data_scaled_dynamic():
    """
    Obtain the dataset from multiple files.
    Returns: X_train, X_test, y_train, y_test
    """
    # Data directory
    DATADIR = 'UCI_HAR_Dataset'
    # 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"
        ]
    from sklearn.base import BaseEstimator, TransformerMixin
    class scaling_tseries_data(BaseEstimator, TransformerMixin):
        from sklearn.preprocessing import StandardScaler
        def __init__(self):
            self.scale = None

        def transform(self, X):
            temp_X1 = X.reshape((X.shape[0] * X.shape[1], X.shape[2]))
```

```
            temp_X1 = self.scale.transform(temp_X1)
            return temp_X1.reshape(X.shape)

        def fit(self, X):
            # remove overlaping
            remove = int(X.shape[1] / 2)
            temp_X = X[:, -remove:, :]
            # flatten data
            temp_X = temp_X.reshape((temp_X.shape[0] * temp_X.shape[1], temp_X.shape[2]))
            scale = StandardScaler()
            scale.fit(temp_X)
            pickle.dump(scale,open('Scale_dynamic.p','wb'))
            self.scale = scale
            return self

    # 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))

    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]
        y_subset = y<=3
        y = y[y_subset]
        return pd.get_dummies(y).as_matrix(),y_subset

    Y_train_d,y_train_sub = load_y('train')
    Y_test_d,y_test_sub = load_y('test')
    X_train_d, X_test_d = load_signals('train'), load_signals('test')
    X_train_d = X_train_d[y_train_sub]
    X_test_d = X_test_d[y_test_sub]

    ###Scling data
    Scale = scaling_tseries_data()
    Scale.fit(X_train_d)
    X_train_d = Scale.transform(X_train_d)
    X_test_d = Scale.transform(X_test_d)

    return X_train_d, Y_train_d, X_test_d,  Y_test_d
```

In [97]:

```
X_train_d, Y_train_d, X_test_d,  Y_test_d = data_scaled_dynamic()
print('Train X shape',X_train_d.shape,'Test X shape',X_test_d.shape)
print('Train Y shape',Y_train_d.shape,'Test Y shape',Y_test_d.shape)
```

```
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-
packages\ipykernel_launcher.py:77: FutureWarning: Method .as_matrix will be removed in a future ve
rsion. Use .values instead.
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-
packages\ipykernel_launcher.py:59: FutureWarning: Method .as_matrix will be removed in a future ve
rsion. Use .values instead.
```

```
Train X shape (3285, 128, 9) Test X shape (1387, 128, 9)
Train Y shape (3285, 3) Test Y shape (1387, 3)
```

In [103]:

```
def model_hyperas(space,verbose=1):
    # Initiliazing the sequential model
    model = Sequential()
    model.add(Conv1D(filters=space['filters'], kernel_size=space['kernel_size'],activation='relu',
                    kernel_initializer='he_uniform',
                    kernel_regularizer=l2(space['l2']),input_shape=(128,9)))
    model.add(Conv1D(filters=space['filters_1'], kernel_size=space['kernel_size_1'],

activation='relu',kernel_regularizer=l2(space['l2_1']),kernel_initializer='he_uniform'))
    model.add(Dropout(space['Dropout']))
    model.add(MaxPooling1D(pool_size=space['pool_size']))
    model.add(Flatten())
    model.add(Dense(space['Dense'], activation='relu'))
    model.add(Dense(3, activation='softmax'))
    adam = optimizers.Adam(lr=space['lr'])
    rmsprop = optimizers.RMSprop(lr=space['lr_1'])
    choiceval = space['choiceval']
    if choiceval == 'adam':
        optim = adam
    else:
        optim = rmsprop
    print(model.summary())
    model.compile(loss='categorical_crossentropy', metrics=['accuracy'],optimizer=optim)
    result = model.fit(X_train_d, Y_train_d,
                    batch_size=space['Dense_1'],
                    nb_epoch=space['nb_epoch'],
                    verbose=verbose,
                    validation_data=(X_test_d, Y_test_d))
    #K.clear_session()
    return model,result
```

In [104]:

```
params_1={'Dense': 64,
 'Dense_1': 32,
 'Dropout': 0.6725241946290972,
 'choiceval': 'adam',
 'filters': 32,
 'filters_1': 32,
 'kernel_size': 7,
 'kernel_size_1': 7,
 'l2': 0.548595947917793,
 'l2_1': 0.28312064960787986,
 'lr': 0.00083263584783479,
 'lr_1': 0.0020986605171288,
 'nb_epoch': 35,
 'pool_size': 5}
best_model,result = model_hyperas(params_1)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv1d_23 (Conv1D)           (None, 122, 32)           2048      
_________________________________________________________________
conv1d_24 (Conv1D)           (None, 116, 32)           7200      
_________________________________________________________________
dropout_12 (Dropout)         (None, 116, 32)           0         
_________________________________________________________________
max_pooling1d_12 (MaxPooling (None, 23, 32)            0         
_________________________________________________________________
flatten_12 (Flatten)         (None, 736)               0         
_________________________________________________________________
dense_23 (Dense)             (None, 64)                47168     
_________________________________________________________________
dense_24 (Dense)             (None, 3)                 195       
=================================================================
Total params: 56,611
Trainable params: 56,611
Non-trainable params: 0
_________________________________________________________________
None
```

```
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-packages\ipykernel_launcher.py:27: UserWarning: The `nb_epoch` argument in `fit` has been renamed `epochs`.
```

```
Train on 3285 samples, validate on 1387 samples
Epoch 1/35
3285/3285 [==============================] - 3s 953us/step - loss: 35.8851 - acc: 0.7227 - val_loss: 21.0044 - val_acc: 0.8875
Epoch 2/35
3285/3285 [==============================] - 2s 626us/step - loss: 12.9971 - acc: 0.9522 - val_loss: 7.6725 - val_acc: 0.8919
Epoch 3/35
3285/3285 [==============================] - 2s 629us/step - loss: 4.5710 - acc: 0.9805 - val_loss: 2.8887 - val_acc: 0.9387
Epoch 4/35
3285/3285 [==============================] - 2s 634us/step - loss: 1.6358 - acc: 0.9839 - val_loss: 1.2601 - val_acc: 0.9438
Epoch 5/35
3285/3285 [==============================] - 2s 644us/step - loss: 0.6514 - acc: 0.9890 - val_loss: 0.7094 - val_acc: 0.9380
Epoch 6/35
3285/3285 [==============================] - 2s 630us/step - loss: 0.3190 - acc: 0.9936 - val_loss: 0.5403 - val_acc: 0.9329
Epoch 7/35
3285/3285 [==============================] - 2s 644us/step - loss: 0.2234 - acc: 0.9878 - val_loss: 0.4513 - val_acc: 0.9618
Epoch 8/35
3285/3285 [==============================] - 2s 634us/step - loss: 0.1841 - acc: 0.9887 - val_loss: 0.4817 - val_acc: 0.9034
Epoch 9/35
3285/3285 [==============================] - 2s 629us/step - loss: 0.1618 - acc: 0.9927 - val_loss: 0.4054 - val_acc: 0.9575
Epoch 10/35
3285/3285 [==============================] - 2s 640us/step - loss: 0.1407 - acc: 0.9939 - val_loss: 0.4076 - val_acc: 0.9402
Epoch 11/35
3285/3285 [==============================] - 2s 689us/step - loss: 0.1447 - acc: 0.9915 - val_loss: 0.3652 - val_acc: 0.9733
Epoch 12/35
3285/3285 [==============================] - 2s 705us/step - loss: 0.1186 - acc: 0.9973 - val_loss: 0.3576 - val_acc: 0.9690
Epoch 13/35
3285/3285 [==============================] - 2s 705us/step - loss: 0.1292 - acc: 0.9921 - val_loss: 0.3495 - val_acc: 0.9611
Epoch 14/35
3285/3285 [==============================] - 2s 721us/step - loss: 0.1189 - acc: 0.9921 - val_loss: 0.4144 - val_acc: 0.8875
Epoch 15/35
3285/3285 [==============================] - 3s 775us/step - loss: 0.1591 - acc: 0.9866 - val_loss: 0.2984 - val_acc: 0.9733
Epoch 16/35
3285/3285 [==============================] - 3s 765us/step - loss: 0.1003 - acc: 0.9982 - val_loss: 0.3098 - val_acc: 0.9697
Epoch 17/35
3285/3285 [==============================] - 2s 731us/step - loss: 0.0903 - acc: 0.9991 - val_loss: 0.3199 - val_acc: 0.9596
Epoch 18/35
3285/3285 [==============================] - 2s 695us/step - loss: 0.1099 - acc: 0.9890 - val_loss: 0.3578 - val_acc: 0.9394
Epoch 19/35
3285/3285 [==============================] - 2s 705us/step - loss: 0.1430 - acc: 0.9881 - val_loss: 0.2951 - val_acc: 0.9582
Epoch 20/35
3285/3285 [==============================] - 2s 715us/step - loss: 0.0891 - acc: 0.9985 - val_loss: 0.2899 - val_acc: 0.9640
Epoch 21/35
3285/3285 [==============================] - 2s 734us/step - loss: 0.0785 - acc: 0.9985 - val_loss: 0.2891 - val_acc: 0.9798
Epoch 22/35
3285/3285 [==============================] - 3s 772us/step - loss: 0.0822 - acc: 0.9963 - val_loss: 0.2834 - val_acc: 0.9683
Epoch 23/35
3285/3285 [==============================] - 2s 715us/step - loss: 0.0722 - acc: 0.9985 - val_loss: 0.3027 - val_acc: 0.9661
Epoch 24/35
3285/3285 [==============================] - 2s 705us/step - loss: 0.0726 - acc: 0.9979 -
```

```
val_loss: 0.2750 - val_acc: 0.9704
Epoch 25/35
3285/3285 [==============================] - 2s 701us/step - loss: 0.1524 - acc: 0.9802 -
val_loss: 0.2676 - val_acc: 0.9531
Epoch 26/35
3285/3285 [==============================] - 3s 815us/step - loss: 0.0870 - acc: 0.9973 -
val_loss: 0.2377 - val_acc: 0.9748
Epoch 27/35
3285/3285 [==============================] - 2s 712us/step - loss: 0.0685 - acc: 0.9991 -
val_loss: 0.2616 - val_acc: 0.9769
Epoch 28/35
3285/3285 [==============================] - 2s 700us/step - loss: 0.0831 - acc: 0.9930 -
val_loss: 0.2876 - val_acc: 0.9589
Epoch 29/35
3285/3285 [==============================] - 2s 710us/step - loss: 0.0969 - acc: 0.9884 -
val_loss: 0.2571 - val_acc: 0.9733
Epoch 30/35
3285/3285 [==============================] - 2s 690us/step - loss: 0.0646 - acc: 0.9994 -
val_loss: 0.2549 - val_acc: 0.9661
Epoch 31/35
3285/3285 [==============================] - 2s 699us/step - loss: 0.0655 - acc: 0.9976 -
val_loss: 0.2418 - val_acc: 0.9726
Epoch 32/35
3285/3285 [==============================] - 2s 701us/step - loss: 0.0636 - acc: 0.9991 -
val_loss: 0.2449 - val_acc: 0.9813
Epoch 33/35
3285/3285 [==============================] - 2s 715us/step - loss: 0.0975 - acc: 0.9887 -
val_loss: 0.3008 - val_acc: 0.9445
Epoch 34/35
3285/3285 [==============================] - 2s 724us/step - loss: 0.1100 - acc: 0.9930 -
val_loss: 0.2731 - val_acc: 0.9301
Epoch 35/35
3285/3285 [==============================] - 3s 767us/step - loss: 0.0612 - acc: 0.9997 -
val_loss: 0.2830 - val_acc: 0.9524
```

In [105]:

```
_,acc_test = best_model.evaluate(X_test_d,Y_test_d,verbose=0)
_,acc_train = best_model.evaluate(X_train_d,Y_train_d,verbose=0)
print('Train_accuracy',acc_train,'test_accuracy',acc_test)
```

```
Train_accuracy 0.9960426179604261 test_accuracy 0.9524152847873107
```

In [116]:

```
params_2={'Dense': 32,
 'Dense_1': 32,
 'Dropout': 0.48642317342570957,
 'choiceval': 'adam',
 'filters': 32,
 'filters_1': 32,
 'kernel_size': 7,
 'kernel_size_1': 7,
 'l2': 0.10401484931072974,
 'l2_1': 0.7228970346142163,
 'lr': 0.000772514731035696,
 'lr_1': 0.003074353392879209,
 'nb_epoch': 70,
 'pool_size': 5}
best_model,result = model_hyperas(params_2)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_29 (Conv1D)           (None, 122, 32)           2048
_________________________________________________________________
conv1d_30 (Conv1D)           (None, 116, 32)           7200
_________________________________________________________________
dropout_15 (Dropout)         (None, 116, 32)           0
_________________________________________________________________
max_pooling1d_15 (MaxPooling (None, 23, 32)            0
_________________________________________________________________
flatten_15 (Flatten)         (None, 736)               0
```

```
_________________________________________________________________
dense_29 (Dense)             (None, 32)                23584
_________________________________________________________________
dense_30 (Dense)             (None, 3)                 99
=================================================================
Total params: 32,931
Trainable params: 32,931
Non-trainable params: 0
_________________________________________________________________
None
```

```
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-
packages\ipykernel_launcher.py:27: UserWarning: The `nb_epoch` argument in `fit` has been renamed
`epochs`.
```

```
Train on 3285 samples, validate on 1387 samples
Epoch 1/70
3285/3285 [==============================] - 3s 923us/step - loss: 31.4792 - acc: 0.6198 -
val_loss: 14.9309 - val_acc: 0.8169
Epoch 2/70
3285/3285 [==============================] - 2s 597us/step - loss: 8.2238 - acc: 0.9297 -
val_loss: 4.3681 - val_acc: 0.8882
Epoch 3/70
3285/3285 [==============================] - 2s 603us/step - loss: 2.5182 - acc: 0.9784 -
val_loss: 1.7604 - val_acc: 0.9221
Epoch 4/70
3285/3285 [==============================] - 2s 614us/step - loss: 1.0404 - acc: 0.9900 -
val_loss: 0.9397 - val_acc: 0.9488
Epoch 5/70
3285/3285 [==============================] - 2s 608us/step - loss: 0.5401 - acc: 0.9942 -
val_loss: 0.6147 - val_acc: 0.9603
Epoch 6/70
3285/3285 [==============================] - 2s 615us/step - loss: 0.3319 - acc: 0.9954 -
val_loss: 0.4636 - val_acc: 0.9690
Epoch 7/70
3285/3285 [==============================] - 2s 598us/step - loss: 0.2339 - acc: 0.9960 -
val_loss: 0.4202 - val_acc: 0.9546
Epoch 8/70
3285/3285 [==============================] - 2s 614us/step - loss: 0.1912 - acc: 0.9951 -
val_loss: 0.4038 - val_acc: 0.9329
Epoch 9/70
3285/3285 [==============================] - 2s 598us/step - loss: 0.1755 - acc: 0.9906 -
val_loss: 0.3564 - val_acc: 0.9661
Epoch 10/70
3285/3285 [==============================] - 2s 603us/step - loss: 0.1458 - acc: 0.9976 -
val_loss: 0.3377 - val_acc: 0.9618
Epoch 11/70
3285/3285 [==============================] - 2s 609us/step - loss: 0.1392 - acc: 0.9982 -
val_loss: 0.3202 - val_acc: 0.9618
Epoch 12/70
3285/3285 [==============================] - 2s 609us/step - loss: 0.1251 - acc: 0.9973 -
val_loss: 0.2956 - val_acc: 0.9683
Epoch 13/70
3285/3285 [==============================] - 2s 608us/step - loss: 0.1149 - acc: 0.9973 -
val_loss: 0.2755 - val_acc: 0.9690
Epoch 14/70
3285/3285 [==============================] - 2s 618us/step - loss: 0.1085 - acc: 0.9982 -
val_loss: 0.2615 - val_acc: 0.9726
Epoch 15/70
3285/3285 [==============================] - 2s 614us/step - loss: 0.1052 - acc: 0.9973 -
val_loss: 0.2698 - val_acc: 0.9596
Epoch 16/70
3285/3285 [==============================] - 2s 608us/step - loss: 0.0995 - acc: 0.9982 -
val_loss: 0.2523 - val_acc: 0.9791
Epoch 17/70
3285/3285 [==============================] - 2s 611us/step - loss: 0.1179 - acc: 0.9893 -
val_loss: 0.2485 - val_acc: 0.9553
Epoch 18/70
3285/3285 [==============================] - 2s 612us/step - loss: 0.0941 - acc: 0.9973 -
val_loss: 0.2351 - val_acc: 0.9762
Epoch 19/70
3285/3285 [==============================] - 2s 608us/step - loss: 0.0824 - acc: 0.9994 -
val_loss: 0.2552 - val_acc: 0.9632
Epoch 20/70
3285/3285 [==============================] - 2s 619us/step - loss: 0.0892 - acc: 0.9976 -
val_loss: 0.2420 - val_acc: 0.9755
```

```
Epoch 21/70
3285/3285 [==============================] - 2s 681us/step - loss: 0.0864 - acc: 0.9973 -
val_loss: 0.2297 - val_acc: 0.9719
Epoch 22/70
3285/3285 [==============================] - 2s 658us/step - loss: 0.0886 - acc: 0.9957 -
val_loss: 0.2427 - val_acc: 0.9560
Epoch 23/70
3285/3285 [==============================] - 2s 669us/step - loss: 0.0720 - acc: 0.9988 -
val_loss: 0.2305 - val_acc: 0.9740
Epoch 24/70
3285/3285 [==============================] - 2s 624us/step - loss: 0.0857 - acc: 0.9960 -
val_loss: 0.2189 - val_acc: 0.9726
Epoch 25/70
3285/3285 [==============================] - 2s 654us/step - loss: 0.0731 - acc: 0.9988 -
val_loss: 0.2552 - val_acc: 0.9373
Epoch 26/70
3285/3285 [==============================] - 2s 640us/step - loss: 0.0687 - acc: 0.9991 -
val_loss: 0.2347 - val_acc: 0.9575
Epoch 27/70
3285/3285 [==============================] - 2s 654us/step - loss: 0.0646 - acc: 0.9991 -
val_loss: 0.2183 - val_acc: 0.9690
Epoch 28/70
3285/3285 [==============================] - 2s 706us/step - loss: 0.0695 - acc: 0.9979 -
val_loss: 0.2037 - val_acc: 0.9733
Epoch 29/70
3285/3285 [==============================] - 2s 648us/step - loss: 0.0903 - acc: 0.9933 -
val_loss: 0.2237 - val_acc: 0.9596
Epoch 30/70
3285/3285 [==============================] - 2s 650us/step - loss: 0.0624 - acc: 0.9991 -
val_loss: 0.2014 - val_acc: 0.9719
Epoch 31/70
3285/3285 [==============================] - 2s 639us/step - loss: 0.0650 - acc: 0.9982 -
val_loss: 0.2109 - val_acc: 0.9567
Epoch 32/70
3285/3285 [==============================] - 2s 639us/step - loss: 0.1057 - acc: 0.9875 -
val_loss: 0.2667 - val_acc: 0.9387
Epoch 33/70
3285/3285 [==============================] - 2s 665us/step - loss: 0.0929 - acc: 0.9924 -
val_loss: 0.2055 - val_acc: 0.9539
Epoch 34/70
3285/3285 [==============================] - 2s 639us/step - loss: 0.0589 - acc: 0.9988 -
val_loss: 0.2269 - val_acc: 0.9553
Epoch 35/70
3285/3285 [==============================] - 2s 639us/step - loss: 0.0560 - acc: 0.9994 -
val_loss: 0.2085 - val_acc: 0.9517
Epoch 36/70
3285/3285 [==============================] - 2s 624us/step - loss: 0.0646 - acc: 0.9963 -
val_loss: 0.1958 - val_acc: 0.9784
Epoch 37/70
3285/3285 [==============================] - 2s 629us/step - loss: 0.0566 - acc: 0.9973 -
val_loss: 0.1749 - val_acc: 0.9661
Epoch 38/70
3285/3285 [==============================] - 2s 634us/step - loss: 0.0546 - acc: 0.9985 -
val_loss: 0.2222 - val_acc: 0.9668
Epoch 39/70
3285/3285 [==============================] - 2s 634us/step - loss: 0.0506 - acc: 0.9994 -
val_loss: 0.2231 - val_acc: 0.9524
Epoch 40/70
3285/3285 [==============================] - 2s 620us/step - loss: 0.0532 - acc: 0.9982 -
val_loss: 0.1891 - val_acc: 0.9697
Epoch 41/70
3285/3285 [==============================] - 2s 634us/step - loss: 0.0499 - acc: 0.9991 -
val_loss: 0.1700 - val_acc: 0.9740
Epoch 42/70
3285/3285 [==============================] - 2s 633us/step - loss: 0.0471 - acc: 0.9991 -
val_loss: 0.1980 - val_acc: 0.9632
Epoch 43/70
3285/3285 [==============================] - 2s 630us/step - loss: 0.0733 - acc: 0.9930 -
val_loss: 0.2057 - val_acc: 0.9438
Epoch 44/70
3285/3285 [==============================] - 2s 633us/step - loss: 0.0600 - acc: 0.9967 -
val_loss: 0.2108 - val_acc: 0.9618
Epoch 45/70
3285/3285 [==============================] - 2s 649us/step - loss: 0.0451 - acc: 0.9994 -
val_loss: 0.1963 - val_acc: 0.9640
Epoch 46/70
3285/3285 [==============================] - 2s 629us/step - loss: 0.0490 - acc: 0.9988 -
```

```
val_loss: 0.1970 - val_acc: 0.9553
Epoch 47/70
3285/3285 [==============================] - 2s 630us/step - loss: 0.0482 - acc: 0.9988 -
val_loss: 0.2138 - val_acc: 0.9560
Epoch 48/70
3285/3285 [==============================] - 2s 649us/step - loss: 0.0434 - acc: 0.9991 -
val_loss: 0.2149 - val_acc: 0.9416
Epoch 49/70
3285/3285 [==============================] - 2s 629us/step - loss: 0.0427 - acc: 1.0000 -
val_loss: 0.1892 - val_acc: 0.9632
Epoch 50/70
3285/3285 [==============================] - 3s 852us/step - loss: 0.0743 - acc: 0.9912 -
val_loss: 0.2000 - val_acc: 0.9387
Epoch 51/70
3285/3285 [==============================] - 2s 655us/step - loss: 0.1378 - acc: 0.9805 -
val_loss: 0.2235 - val_acc: 0.9748
Epoch 52/70
3285/3285 [==============================] - 2s 679us/step - loss: 0.0660 - acc: 0.9985 -
val_loss: 0.1799 - val_acc: 0.9740
Epoch 53/70
3285/3285 [==============================] - 2s 719us/step - loss: 0.0450 - acc: 0.9997 -
val_loss: 0.1942 - val_acc: 0.9654
Epoch 54/70
3285/3285 [==============================] - 3s 767us/step - loss: 0.0413 - acc: 1.0000 -
val_loss: 0.1731 - val_acc: 0.9755
Epoch 55/70
3285/3285 [==============================] - 3s 809us/step - loss: 0.0511 - acc: 0.9957 -
val_loss: 0.1939 - val_acc: 0.9603
Epoch 56/70
3285/3285 [==============================] - 3s 768us/step - loss: 0.0602 - acc: 0.9967 -
val_loss: 0.1920 - val_acc: 0.9618
Epoch 57/70
3285/3285 [==============================] - 2s 696us/step - loss: 0.0405 - acc: 0.9997 -
val_loss: 0.2638 - val_acc: 0.9409
Epoch 58/70
3285/3285 [==============================] - 2s 640us/step - loss: 0.0400 - acc: 0.9997 -
val_loss: 0.2365 - val_acc: 0.9503
Epoch 59/70
3285/3285 [==============================] - 2s 650us/step - loss: 0.0390 - acc: 0.9988 -
val_loss: 0.1714 - val_acc: 0.9690
Epoch 60/70
3285/3285 [==============================] - 2s 639us/step - loss: 0.0581 - acc: 0.9927 -
val_loss: 0.2328 - val_acc: 0.9257
Epoch 61/70
3285/3285 [==============================] - 2s 638us/step - loss: 0.1069 - acc: 0.9826 -
val_loss: 0.2174 - val_acc: 0.9740
Epoch 62/70
3285/3285 [==============================] - 2s 641us/step - loss: 0.0515 - acc: 0.9991 -
val_loss: 0.1936 - val_acc: 0.9625
Epoch 63/70
3285/3285 [==============================] - 2s 654us/step - loss: 0.0428 - acc: 0.9994 -
val_loss: 0.1904 - val_acc: 0.9668
Epoch 64/70
3285/3285 [==============================] - 2s 654us/step - loss: 0.0376 - acc: 1.0000 -
val_loss: 0.1900 - val_acc: 0.9632
Epoch 65/70
3285/3285 [==============================] - 2s 635us/step - loss: 0.0358 - acc: 0.9997 -
val_loss: 0.1854 - val_acc: 0.9582
Epoch 66/70
3285/3285 [==============================] - 4s 1ms/step - loss: 0.0349 - acc: 1.0000 - val_loss:
0.1841 - val_acc: 0.9676
Epoch 67/70
3285/3285 [==============================] - 2s 714us/step - loss: 0.0339 - acc: 0.9997 -
val_loss: 0.1796 - val_acc: 0.9647
Epoch 68/70
3285/3285 [==============================] - 3s 783us/step - loss: 0.0339 - acc: 0.9997 -
val_loss: 0.1841 - val_acc: 0.9632
Epoch 69/70
3285/3285 [==============================] - 2s 685us/step - loss: 0.0374 - acc: 0.9994 -
val_loss: 0.1656 - val_acc: 0.9553
Epoch 70/70
3285/3285 [==============================] - 2s 671us/step - loss: 0.0595 - acc: 0.9945 -
val_loss: 0.2281 - val_acc: 0.9402
```

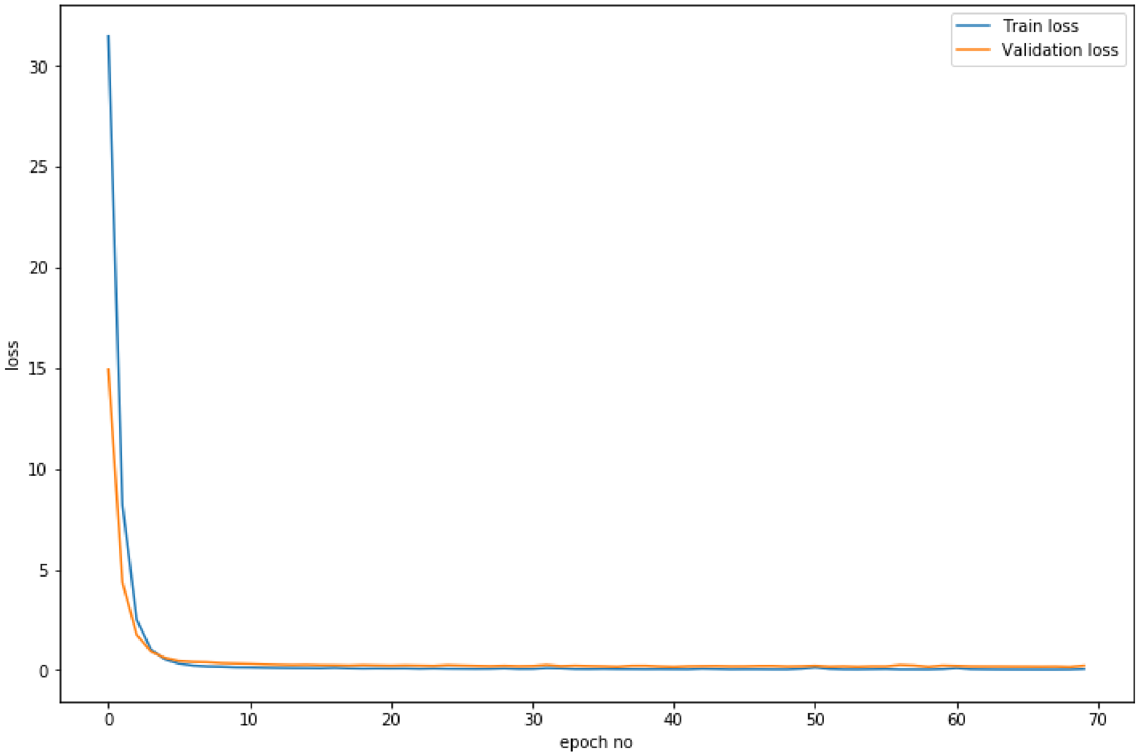
In [118]:

```
_,acc_test = best_model.evaluate(X_test_d,Y_test_d,verbose=0)
_,acc_train = best_model.evaluate(X_train_d,Y_train_d,verbose=0)
print('Train_accuracy',acc_train,'test_accuracy',acc_test)
```

```
Train_accuracy 0.9990867579908675 test_accuracy 0.9401586157173756
```

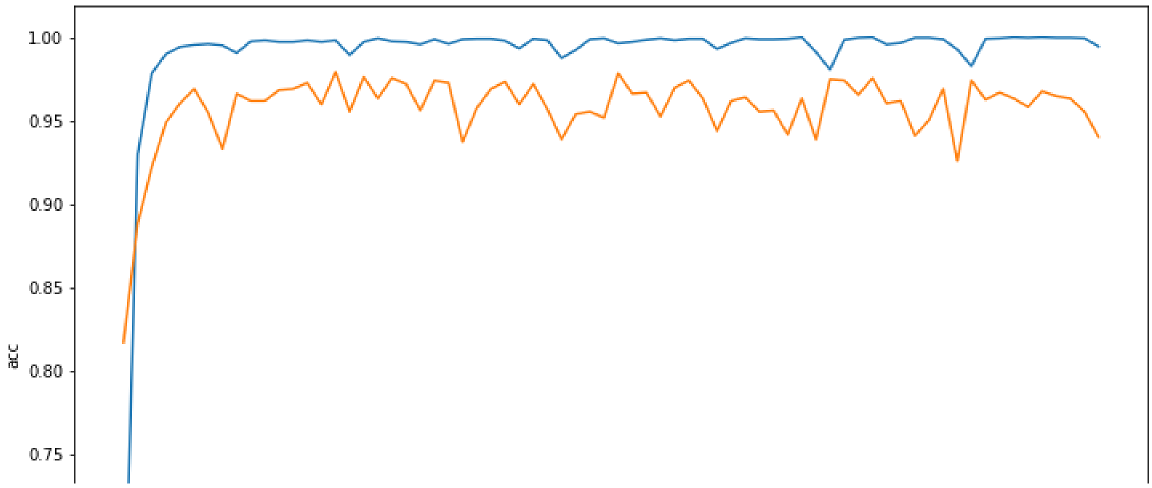
In [119]:

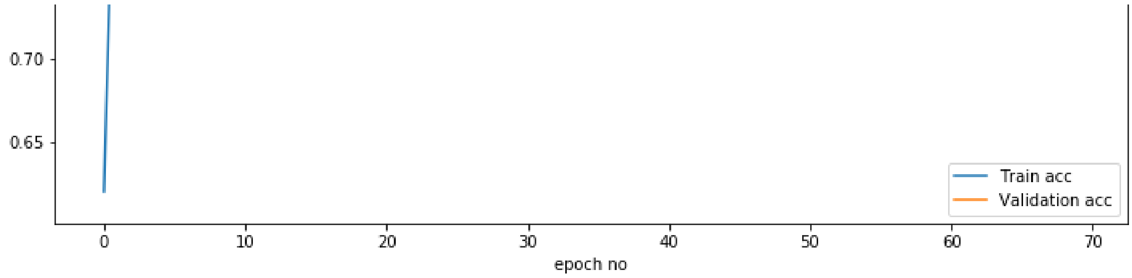
```
plt.figure(figsize=(12,8))
plt.plot(result.history['loss'],label='Train loss')
plt.plot(result.history['val_loss'],label = 'Validation loss')
plt.xlabel('epoch no')
plt.ylabel('loss')
plt.legend()
plt.show()
```




In [120]:

```
plt.figure(figsize=(12,8))
plt.plot(result.history['acc'],label='Train acc')
plt.plot(result.history['val_acc'],label = 'Validation acc')
plt.xlabel('epoch no')
plt.ylabel('acc')
plt.legend()
plt.show()
```






In [121]:

```
params_3={'Dense': 32,
 'Dense_1': 32,
 'Dropout': 0.48642317342570957,
 'choiceval': 'adam',
 'filters': 32,
 'filters_1': 32,
 'kernel_size': 7,
 'kernel_size_1': 7,
 'l2': 0.10401484931072974,
 'l2_1': 0.7228970346142163,
 'lr': 0.000772514731035696,
 'lr_1': 0.003074353392879209,
 'nb_epoch': 54,
 'pool_size': 5}
best_model,result = model_hyperas(params_3)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_31 (Conv1D)           (None, 122, 32)           2048
_________________________________________________________________
conv1d_32 (Conv1D)           (None, 116, 32)           7200
_________________________________________________________________
dropout_16 (Dropout)         (None, 116, 32)           0
_________________________________________________________________
max_pooling1d_16 (MaxPooling (None, 23, 32)            0
_________________________________________________________________
flatten_16 (Flatten)         (None, 736)               0
_________________________________________________________________
dense_31 (Dense)             (None, 32)                23584
_________________________________________________________________
dense_32 (Dense)             (None, 3)                 99
=================================================================
Total params: 32,931
Trainable params: 32,931
Non-trainable params: 0
_________________________________________________________________
None
```

```
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-
packages\ipykernel_launcher.py:27: UserWarning: The `nb_epoch` argument in `fit` has been renamed
`epochs`.
```

```
Train on 3285 samples, validate on 1387 samples
Epoch 1/54
3285/3285 [==============================] - 3s 945us/step - loss: 31.9222 - acc: 0.6228 -
val_loss: 15.2623 - val_acc: 0.7318
Epoch 2/54
3285/3285 [==============================] - 2s 641us/step - loss: 8.4427 - acc: 0.8962 -
val_loss: 4.4591 - val_acc: 0.8983
Epoch 3/54
3285/3285 [==============================] - 2s 629us/step - loss: 2.5809 - acc: 0.9689 -
val_loss: 1.7617 - val_acc: 0.9351
Epoch 4/54
3285/3285 [==============================] - 2s 685us/step - loss: 1.0455 - acc: 0.9860 -
val_loss: 0.9438 - val_acc: 0.9438
Epoch 5/54
3285/3285 [==============================] - 2s 639us/step - loss: 0.5311 - acc: 0.9915 -
val_loss: 0.6133 - val_acc: 0.9524
Epoch 6/54
3285/3285 [==============================] - 2s 635us/step - loss: 0.3267 - acc: 0.9918 -
```

```
val_loss: 0.4697 - val_acc: 0.9560
Epoch 7/54
3285/3285 [==============================] - 2s 698us/step - loss: 0.2242 - acc: 0.9948 -
val_loss: 0.3867 - val_acc: 0.9683
Epoch 8/54
3285/3285 [==============================] - 2s 633us/step - loss: 0.1798 - acc: 0.9948 -
val_loss: 0.3515 - val_acc: 0.9618
Epoch 9/54
3285/3285 [==============================] - 2s 636us/step - loss: 0.1497 - acc: 0.9960 -
val_loss: 0.3277 - val_acc: 0.9618
Epoch 10/54
3285/3285 [==============================] - 2s 684us/step - loss: 0.1311 - acc: 0.9988 -
val_loss: 0.3009 - val_acc: 0.9661
Epoch 11/54
3285/3285 [==============================] - 2s 637us/step - loss: 0.1239 - acc: 0.9973 -
val_loss: 0.2939 - val_acc: 0.9589
Epoch 12/54
3285/3285 [==============================] - 2s 648us/step - loss: 0.1159 - acc: 0.9970 -
val_loss: 0.2900 - val_acc: 0.9603
Epoch 13/54
3285/3285 [==============================] - 2s 669us/step - loss: 0.1102 - acc: 0.9976 -
val_loss: 0.3657 - val_acc: 0.9005
Epoch 14/54
3285/3285 [==============================] - 2s 659us/step - loss: 0.1294 - acc: 0.9915 -
val_loss: 0.2559 - val_acc: 0.9755
Epoch 15/54
3285/3285 [==============================] - 2s 642us/step - loss: 0.1088 - acc: 0.9963 -
val_loss: 0.2997 - val_acc: 0.9322
Epoch 16/54
3285/3285 [==============================] - 2s 641us/step - loss: 0.1284 - acc: 0.9893 -
val_loss: 0.2395 - val_acc: 0.9712
Epoch 17/54
3285/3285 [==============================] - 2s 655us/step - loss: 0.0906 - acc: 0.9994 -
val_loss: 0.2807 - val_acc: 0.9517
Epoch 18/54
3285/3285 [==============================] - 2s 665us/step - loss: 0.0895 - acc: 0.9985 -
val_loss: 0.2569 - val_acc: 0.9625
Epoch 19/54
3285/3285 [==============================] - 2s 725us/step - loss: 0.0876 - acc: 0.9976 -
val_loss: 0.2327 - val_acc: 0.9740
Epoch 20/54
3285/3285 [==============================] - 2s 671us/step - loss: 0.0761 - acc: 0.9997 -
val_loss: 0.2179 - val_acc: 0.9733
Epoch 21/54
3285/3285 [==============================] - 2s 679us/step - loss: 0.0763 - acc: 0.9994 -
val_loss: 0.2680 - val_acc: 0.9524
Epoch 22/54
3285/3285 [==============================] - 2s 660us/step - loss: 0.1238 - acc: 0.9848 -
val_loss: 0.2747 - val_acc: 0.9366
Epoch 23/54
3285/3285 [==============================] - 2s 679us/step - loss: 0.1019 - acc: 0.9924 -
val_loss: 0.2483 - val_acc: 0.9611
Epoch 24/54
3285/3285 [==============================] - 2s 670us/step - loss: 0.0748 - acc: 0.9985 -
val_loss: 0.2016 - val_acc: 0.9697
Epoch 25/54
3285/3285 [==============================] - 2s 686us/step - loss: 0.0688 - acc: 0.9994 -
val_loss: 0.2091 - val_acc: 0.9697
Epoch 26/54
3285/3285 [==============================] - 2s 674us/step - loss: 0.0799 - acc: 0.9948 -
val_loss: 0.2191 - val_acc: 0.9596
Epoch 27/54
3285/3285 [==============================] - 2s 690us/step - loss: 0.0779 - acc: 0.9970 -
val_loss: 0.2083 - val_acc: 0.9755
Epoch 28/54
3285/3285 [==============================] - 2s 680us/step - loss: 0.0626 - acc: 0.9997 -
val_loss: 0.2093 - val_acc: 0.9755
Epoch 29/54
3285/3285 [==============================] - 2s 673us/step - loss: 0.0802 - acc: 0.9945 -
val_loss: 0.2298 - val_acc: 0.9769
Epoch 30/54
3285/3285 [==============================] - 2s 670us/step - loss: 0.0726 - acc: 0.9976 -
val_loss: 0.1776 - val_acc: 0.9791
Epoch 31/54
3285/3285 [==============================] - 2s 680us/step - loss: 0.0626 - acc: 0.9988 -
val_loss: 0.2208 - val_acc: 0.9661
Epoch 32/54
```

```
3285/3285 [==============================] - 2s 674us/step - loss: 0.0574 - acc: 0.9991 -
val_loss: 0.1994 - val_acc: 0.9712
Epoch 33/54
3285/3285 [==============================] - 2s 684us/step - loss: 0.0544 - acc: 1.0000 -
val_loss: 0.2063 - val_acc: 0.9755
Epoch 34/54
3285/3285 [==============================] - 2s 671us/step - loss: 0.0837 - acc: 0.9921 -
val_loss: 0.1779 - val_acc: 0.9733
Epoch 35/54
3285/3285 [==============================] - 3s 797us/step - loss: 0.0767 - acc: 0.9954 -
val_loss: 0.1845 - val_acc: 0.9683
Epoch 36/54
3285/3285 [==============================] - 2s 684us/step - loss: 0.0549 - acc: 0.9997 -
val_loss: 0.2147 - val_acc: 0.9654
Epoch 37/54
3285/3285 [==============================] - 2s 691us/step - loss: 0.0550 - acc: 0.9982 -
val_loss: 0.1805 - val_acc: 0.9661
Epoch 38/54
3285/3285 [==============================] - 2s 679us/step - loss: 0.0536 - acc: 0.9988 -
val_loss: 0.2387 - val_acc: 0.9531
Epoch 39/54
3285/3285 [==============================] - 2s 744us/step - loss: 0.0520 - acc: 0.9991 -
val_loss: 0.1893 - val_acc: 0.9748
Epoch 40/54
3285/3285 [==============================] - 2s 702us/step - loss: 0.0477 - acc: 1.0000 -
val_loss: 0.1664 - val_acc: 0.9805
Epoch 41/54
3285/3285 [==============================] - 2s 711us/step - loss: 0.0524 - acc: 0.9991 -
val_loss: 0.2059 - val_acc: 0.9531
Epoch 42/54
3285/3285 [==============================] - 2s 685us/step - loss: 0.0717 - acc: 0.9933 -
val_loss: 0.2707 - val_acc: 0.9135
Epoch 43/54
3285/3285 [==============================] - 2s 684us/step - loss: 0.1186 - acc: 0.9848 -
val_loss: 0.2431 - val_acc: 0.9510
Epoch 44/54
3285/3285 [==============================] - 2s 714us/step - loss: 0.0583 - acc: 0.9985 -
val_loss: 0.1792 - val_acc: 0.9748
Epoch 45/54
3285/3285 [==============================] - 2s 700us/step - loss: 0.0460 - acc: 0.9997 -
val_loss: 0.1889 - val_acc: 0.9776
Epoch 46/54
3285/3285 [==============================] - 3s 809us/step - loss: 0.0437 - acc: 1.0000 -
val_loss: 0.1784 - val_acc: 0.9769
Epoch 47/54
3285/3285 [==============================] - 2s 689us/step - loss: 0.0442 - acc: 0.9997 -
val_loss: 0.1903 - val_acc: 0.9654
Epoch 48/54
3285/3285 [==============================] - 2s 675us/step - loss: 0.0425 - acc: 1.0000 -
val_loss: 0.2280 - val_acc: 0.9640
Epoch 49/54
3285/3285 [==============================] - 2s 710us/step - loss: 0.0456 - acc: 0.9994 -
val_loss: 0.1745 - val_acc: 0.9733
Epoch 50/54
3285/3285 [==============================] - 2s 695us/step - loss: 0.0447 - acc: 0.9988 -
val_loss: 0.1524 - val_acc: 0.971924 -
Epoch 51/54
3285/3285 [==============================] - 2s 685us/step - loss: 0.0576 - acc: 0.9963 -
val_loss: 0.2544 - val_acc: 0.9589
Epoch 52/54
3285/3285 [==============================] - 2s 677us/step - loss: 0.0479 - acc: 0.9988 -
val_loss: 0.2369 - val_acc: 0.9366
Epoch 53/54
3285/3285 [==============================] - 2s 703us/step - loss: 0.0695 - acc: 0.9921 -
val_loss: 0.2728 - val_acc: 0.9546
Epoch 54/54
3285/3285 [==============================] - 2s 710us/step - loss: 0.0608 - acc: 0.9967 -
val_loss: 0.1644 - val_acc: 0.9755
```

In [122]:

```
_,acc_test = best_model.evaluate(X_test_d,Y_test_d,verbose=0)
_,acc_train = best_model.evaluate(X_train_d,Y_train_d,verbose=0)
print('Train_accuracy',acc_train,'test_accuracy',acc_test)
```

```
Train_accuracy 0.9996955859969558 test_accuracy 0.9754866618601298
```

In [123]:

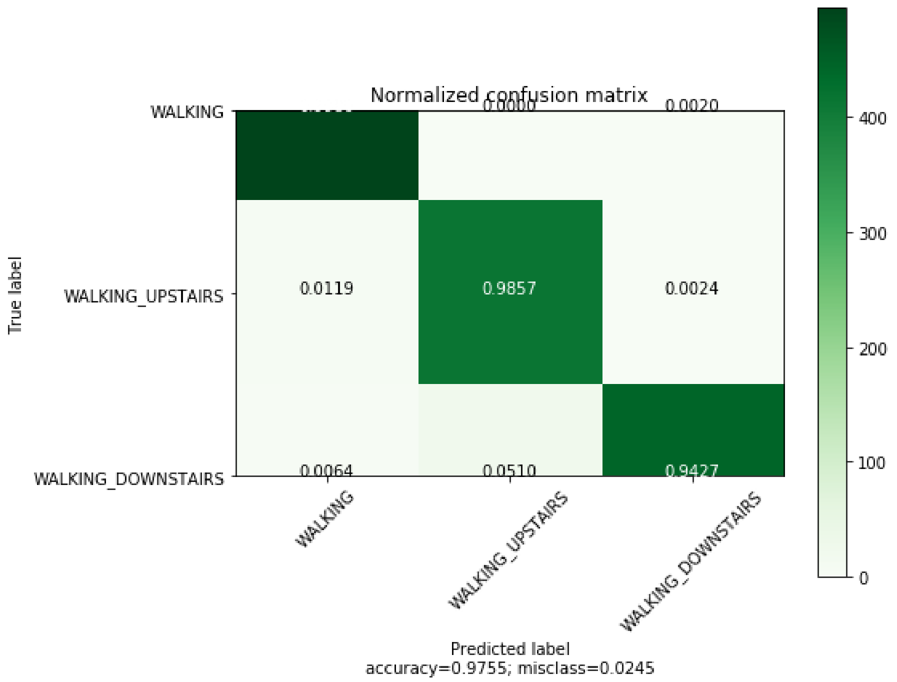
```python
ACTIVITIES = {
    0: 'WALKING',
    1: 'WALKING_UPSTAIRS',
    2: 'WALKING_DOWNSTAIRS',
}

# Utility function to print the confusion matrix
def confusion_matrix_cnn(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'])
    return metrics.confusion_matrix(Y_true, Y_pred)

# Confusion Matrix
print(confusion_matrix_cnn(Y_test_d, best_model.predict(X_test_d)))
```

```
[[495   0   1]
 [  5 414   1]
 [  3  24 444]]
```

In [124]:

```python
plt.figure(figsize=(8,8))
cm = confusion_matrix_cnn(Y_test_d, best_model.predict(X_test_d))
plot_confusion_matrix(cm, target_names=['WALKING','WALKING_UPSTAIRS','WALKING_DOWNSTAIRS'],
                      normalize=True, title='Normalized confusion matrix', cmap = plt.cm.Greens)
plt.show()
```

```
&lt;Figure size 576x576 with 0 Axes&gt;
```




In [125]:

```python
#saving model
best_model.save('final_model_dynamic.h5')
```

In [126]:

```python
def data():
    """
    Obtain the dataset from multiple files.
```

```
    Returns: X_train, X_test, y_train, y_test
    """
    # Data directory
    DATADIR = 'UCI_HAR_Dataset'
    # 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"
        ]
    # 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))

    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 y

    X_train, X_test = load_signals('train'), load_signals('test')
    Y_train, Y_test = load_y('train'), load_y('test')

    return X_train, Y_train, X_test,  Y_test
```

In [127]:

```
X_train, Y_train, X_test,  Y_test = data()
print('shape of test Y',Y_test.shape)
```

```
C:\Users\mchetankumar\AppData\Local\Continuum\anaconda3\envs\TaxiEnv\lib\site-
packages\ipykernel_launcher.py:35: FutureWarning: Method .as_matrix will be removed in a future ve
rsion. Use .values instead.
```

```
shape of test Y (2947,)
```

In [128]:

```
from keras.models import load_model

model_2class = load_model('final_model_2class.h5')
model_dynamic = load_model('final_model_dynamic.h5')
model_static = load_model('final_model_static.h5')
scale_2class = pickle.load(open('Scale_2class.p','rb'))
```

```
scale_static = pickle.load(open('Scale_static.p','rb'))
scale_dynamic = pickle.load(open('Scale_dynamic.p','rb'))
```

In [129]:

```
def transform_data(X,scale):
    X_temp = X.reshape((X.shape[0] * X.shape[1], X.shape[2]))
    X_temp = scale.transform(X_temp)
    return X_temp.reshape(X.shape)
```

In [130]:

```
def predict_activity(X):
    ##predicting whether dynamic or static
    predict_2class = model_2class.predict(transform_data(X,scale_2class))
    Y_pred_2class =  np.argmax(predict_2class, axis=1)
    #static data filter
    X_static = X[Y_pred_2class==1]
    #dynamic data filter
    X_dynamic = X[Y_pred_2class==0]
    #predicting static activities
    predict_static = model_static.predict(transform_data(X_static,scale_static))
    predict_static = np.argmax(predict_static,axis=1)
    #adding 4 because need to get inal prediction lable as output
    predict_static = predict_static + 4
    #predicting dynamic activites
    predict_dynamic = model_dynamic.predict(transform_data(X_dynamic,scale_dynamic))
    predict_dynamic = np.argmax(predict_dynamic,axis=1)
    #adding 1 because need to get inal prediction lable as output
    predict_dynamic = predict_dynamic + 1
    ##appending final output to one list in the same sequence of input data
    i,j = 0,0
    final_pred = []
    for mask in Y_pred_2class:
        if mask == 1:
            final_pred.append(predict_static[i])
            i = i + 1
        else:
            final_pred.append(predict_dynamic[j])
            j = j + 1
    return final_pred
```

In [131]:

```
final_pred_test = predict_activity(X_test)
final_pred_train = predict_activity(X_train)
```

In [132]:

```
from sklearn.metrics import accuracy_score
print('Accuracy of train data',accuracy_score(Y_train,final_pred_train))
print('Accuracy of validation data',accuracy_score(Y_test,final_pred_test))
```

```
Accuracy of train data 0.9782372143634385
Accuracy of validation data 0.9616559212758737
```

In [134]:

```
cm = metrics.confusion_matrix(Y_test, final_pred_test,labels=range(1,7))
cm
```

Out[134]:

```
array([[495,   1,   0,   0,   0,   0],
       [  3, 444,  24,   0,   0,   0],
       [  5,   1, 414,   0,   0,   0],
       [  0,  22,   0, 443,  26,   0],
       [  0,   2,   0,  28, 501,   1],
       [  0,   0,   0,   0,   0, 537]], dtype=int64)
```

In [135]:

```
plt.figure(figsize=(8,8))
labels=['WALKING','WALKING_UPSTAIRS','WALKING_DOWNSTAIRS','SITTING','STANDING','LAYING']
plot_confusion_matrix(cm, target_names=labels,
                      normalize=True, title='Normalized confusion matrix', cmap = plt.cm.Greens)
plt.show()
```

<Figure size 576x576 with 0 Axes>