In [1]:

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

import pandas as pd
import numpy as np
```

In [2]:

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

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

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

Data

In [3]:

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

In [4]:

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

Loading X_train and X_test

In [5]:

```
#Loading the train data for X_train
tr_1 = 'UCI_HAR_Dataset/train/Inertial Signals/body_acc_x_train.txt'
tr_2 = 'UCI_HAR_Dataset/train/Inertial Signals/body_acc_y_train.txt'
tr_3 = 'UCI_HAR_Dataset/train/Inertial Signals/body_acc_z_train.txt'
tr_4 = 'UCI_HAR_Dataset/train/Inertial Signals/body_gyro_x_train.txt'
tr_5 = 'UCI_HAR_Dataset/train/Inertial Signals/body_gyro_y_train.txt'
tr_6 = 'UCI_HAR_Dataset/train/Inertial Signals/body_gyro_z_train.txt'
tr_7 = 'UCI_HAR_Dataset/train/Inertial Signals/total_acc_x_train.txt'
tr_8 = 'UCI_HAR_Dataset/train/Inertial Signals/total_acc_y_train.txt'
tr_9 = 'UCI_HAR_Dataset/train/Inertial Signals/total_acc_z_train.txt'
```

In [6]:

```
signals_data = []

signals_data.append(_read_csv(tr_1).as_matrix())
signals_data.append(_read_csv(tr_2).as_matrix())
signals_data.append(_read_csv(tr_3).as_matrix())
signals_data.append(_read_csv(tr_4).as_matrix())
signals_data.append(_read_csv(tr_5).as_matrix())
signals_data.append(_read_csv(tr_6).as_matrix())
signals_data.append(_read_csv(tr_7).as_matrix())
```

```
signals_data.append(_read_csv(tr_7).as_matrix())
signals_data.append(_read_csv(tr_8).as_matrix())
signals_data.append(_read_csv(tr_9).as_matrix())
```

In [7]:

```
# 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)
X_train = np.transpose(signals_data, (1, 2, 0))
```

In [8]:

```
print(len(X_train))
print(len(X_train[0]))
print(len(X_train[0][0]))
```

```
7352
128
9
```

In [9]:

```
#Loading the train data for X_test
ts_1 = 'UCI_HAR_Dataset/test/Inertial Signals/body_acc_x_test.txt'
ts_2 = 'UCI_HAR_Dataset/test/Inertial Signals/body_acc_y_test.txt'
ts_3 = 'UCI_HAR_Dataset/test/Inertial Signals/body_acc_z_test.txt'
ts_4 = 'UCI_HAR_Dataset/test/Inertial Signals/body_gyro_x_test.txt'
ts_5 = 'UCI_HAR_Dataset/test/Inertial Signals/body_gyro_y_test.txt'
ts_6 = 'UCI_HAR_Dataset/test/Inertial Signals/body_gyro_z_test.txt'
ts_7 = 'UCI_HAR_Dataset/test/Inertial Signals/total_acc_x_test.txt'
ts_8 = 'UCI_HAR_Dataset/test/Inertial Signals/total_acc_y_test.txt'
ts_9 = 'UCI_HAR_Dataset/test/Inertial Signals/total_acc_z_test.txt'
```

In [10]:

```
Xtest_signalsdata = []

Xtest_signalsdata.append(_read_csv(ts_1).as_matrix())
Xtest_signalsdata.append(_read_csv(ts_2).as_matrix())
Xtest_signalsdata.append(_read_csv(ts_3).as_matrix())
Xtest_signalsdata.append(_read_csv(ts_4).as_matrix())
Xtest_signalsdata.append(_read_csv(ts_5).as_matrix())
Xtest_signalsdata.append(_read_csv(ts_6).as_matrix())
Xtest_signalsdata.append(_read_csv(ts_7).as_matrix())
Xtest_signalsdata.append(_read_csv(ts_8).as_matrix())
Xtest_signalsdata.append(_read_csv(ts_9).as_matrix())
```

In [11]:

```
X_test = np.transpose(Xtest_signalsdata, (1, 2, 0))
```

In [12]:

```
print(len(X_test))
print(len(X_test[0]))
print(len(X_test[0][0]))
```

```
2947
128
9
```

Loading y_train and y_test

In [13]:

```
#Loading the train data for y_train
y_tr = 'UCI_HAR_Dataset/train/y_train.txt'
```

In [14]:

```
y = _read_csv(y_tr)[0]
```

In [15]:

```
y_train = pd.get_dummies(y).as_matrix()
```

In [16]:

```
len(y_train)
```

Out[16]:

```
7352
```

In [17]:

```
#Loading the train data for y_test
y_tst = 'UCI_HAR_Dataset/test/y_test.txt'
```

In [18]:

```
y_tst = _read_csv(y_tst)[0]
```

In [19]:

```
y_test = pd.get_dummies(y_tst).as_matrix()
```

In [20]:

```
len(y_test)
```

Out[20]:

```
2947
```

In [21]:

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

In [22]:

```
# Configuring a session
session_conf = tf.ConfigProto(
    intra_op_parallelism_threads=1,
    inter_op_parallelism_threads=1
)
```

In [23]:

```
# 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 [25]:

```
# Importing libraries
```

```
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers.core import Dense, Dropout
```

In [26]:

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

In [27]:

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

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

```
128
9
7352
```

1. LSTM with one layer

In [28]:

```
# Initializing parameters
epochs = 30
batch_size = 16
n_hidden = 32
```

In [29]:

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

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
lstm_1 (LSTM)                (None, 32)                5376      
_________________________________________________________________
dropout_1 (Dropout)          (None, 32)                0         
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 198       
=================================================================
Total params: 5,574
Trainable params: 5,574
Non-trainable params: 0
_________________________________________________________________
```

In [30]:

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

In [40]:

```python
# Training the model
model_1 = model.fit(X_train, y_train, batch_size=batch_size, validation_data=(X_test, y_test), epoc
hs=epochs)
score = model.evaluate(X_test, y_test)
print("Accuracy: %.2f%%" % (score[1]*100))
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 53s 7ms/step - loss: 0.1677 - acc: 0.9491 - val_loss:
0.3217 - val_acc: 0.9108
Epoch 2/30
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1604 - acc: 0.9494 - val_loss:
0.4382 - val_acc: 0.9084
Epoch 3/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1540 - acc: 0.9494 - val_loss:
0.4220 - val_acc: 0.9131
Epoch 4/30
7352/7352 [==============================] - 51s 7ms/step - loss: 0.1719 - acc: 0.9478 - val_loss:
0.4104 - val_acc: 0.9182
Epoch 5/30
7352/7352 [==============================] - 53s 7ms/step - loss: 0.1446 - acc: 0.9483 - val_loss:
0.3718 - val_acc: 0.9101
Epoch 6/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1451 - acc: 0.9489 - val_loss:
0.4547 - val_acc: 0.8975
Epoch 7/30
7352/7352 [==============================] - 53s 7ms/step - loss: 0.1480 - acc: 0.9501 - val_loss:
0.4265 - val_acc: 0.9050
Epoch 8/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.1359 - acc: 0.9513 - val_loss:
0.3677 - val_acc: 0.9121
Epoch 9/30
7352/7352 [==============================] - 53s 7ms/step - loss: 0.1453 - acc: 0.9520 - val_loss:
0.4181 - val_acc: 0.9101
Epoch 10/30
7352/7352 [==============================] - 53s 7ms/step - loss: 0.1444 - acc: 0.9523 - val_loss:
0.4241 - val_acc: 0.9141
Epoch 11/30
7352/7352 [==============================] - 53s 7ms/step - loss: 0.1509 - acc: 0.9521 - val_loss:
0.4533 - val_acc: 0.9023
Epoch 12/30
7352/7352 [==============================] - 52s 7ms/step - loss: 0.1568 - acc: 0.9487 - val_loss:
0.4120 - val_acc: 0.9128
Epoch 13/30
7352/7352 [==============================] - 53s 7ms/step - loss: 0.1359 - acc: 0.9505 - val_loss:
0.4156 - val_acc: 0.9155
Epoch 14/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.1395 - acc: 0.9505 - val_loss:
0.5503 - val_acc: 0.8989
Epoch 15/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.1367 - acc: 0.9517 - val_loss:
0.5409 - val_acc: 0.9074
Epoch 16/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.1450 - acc: 0.9524 - val_loss:
0.4455 - val_acc: 0.9121
Epoch 17/30
7352/7352 [==============================] - 55s 7ms/step - loss: 0.1822 - acc: 0.9455 - val_loss:
0.4762 - val_acc: 0.9094
Epoch 18/30
7352/7352 [==============================] - 56s 8ms/step - loss: 0.1501 - acc: 0.9512 - val_loss:
0.4533 - val_acc: 0.9087
Epoch 19/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.1755 - acc: 0.9498 - val_loss:
0.6110 - val_acc: 0.8979
Epoch 20/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.1458 - acc: 0.9517 - val_loss:
0.4519 - val_acc: 0.9114
Epoch 21/30
7352/7352 [==============================] - 55s 7ms/step - loss: 0.1441 - acc: 0.9529 - val_loss:
0.4319 - val_acc: 0.9101
Epoch 22/30
7352/7352 [==============================] - 55s 7ms/step - loss: 0.1351 - acc: 0.9528 - val_loss:
0.4282 - val_acc: 0.9128
Epoch 23/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.1359 - acc: 0.9516 - val_loss:
0.4525 - val_acc: 0.9165
Epoch 24/30
```

```
Epoch 24/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.1528 - acc: 0.9513 - val_loss:
0.5694 - val_acc: 0.9030
Epoch 25/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.3056 - acc: 0.9090 - val_loss:
0.5275 - val_acc: 0.8721
Epoch 26/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.4060 - acc: 0.8896 - val_loss:
0.3637 - val_acc: 0.8992
Epoch 27/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.2385 - acc: 0.9309 - val_loss:
0.3289 - val_acc: 0.9063
Epoch 28/30
7352/7352 [==============================] - 55s 7ms/step - loss: 0.2535 - acc: 0.9217 - val_loss:
0.3733 - val_acc: 0.9043
Epoch 29/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.1883 - acc: 0.9323 - val_loss:
0.3609 - val_acc: 0.9179
Epoch 30/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.1489 - acc: 0.9478 - val_loss:
0.4445 - val_acc: 0.9148
2947/2947 [==============================] - 2s 521us/step
Accuracy: 91.48%
```

In [41]:

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

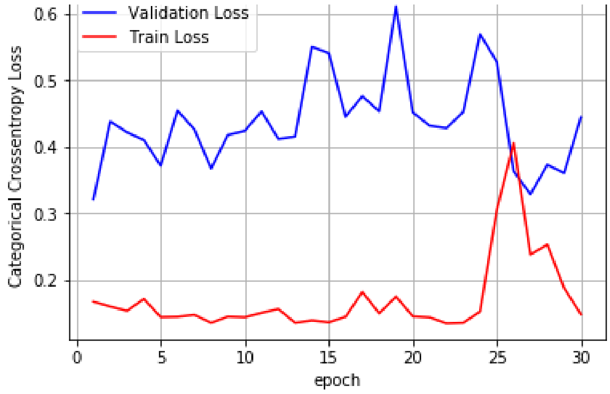
```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 537        0         0        0                   0
SITTING                  0      418        72        0                   1
STANDING                 0      111       420        1                   0
WALKING                  0        0         0      472                  24
WALKING_DOWNSTAIRS       0        0         0        9                 407
WALKING_UPSTAIRS         0        0         0       28                   1

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

In [34]:

```
import matplotlib.pyplot as plt
import time

#this function is used to update the plots for each epoch and error
def plt_dynamic(x, vy, ty, ax, colors=['b']):
    ax.plot(x, vy, 'b', label="Validation Loss")
    ax.plot(x, ty, 'r', label="Train Loss")
    plt.legend()
    plt.grid()
    fig.canvas.draw()
```

In [43]:

```
fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch')
ax.set_ylabel('Categorical Crossentropy Loss')

x = list(range(1,31))

vy = model_1.history['val_loss']
ty = model_1.history['loss']
plt_dynamic(x, vy, ty, ax)
```



2. LSTM with one layer(48 units)

In [30]:

```
model = Sequential()
model.add(LSTM(48, input_shape=(timesteps, input_dim)))
model.add(Dropout(0.5))
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()

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

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
lstm_2 (LSTM)                (None, 48)                11136     
_________________________________________________________________
dropout_2 (Dropout)          (None, 48)                0         
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 294       
=================================================================
Total params: 11,430
Trainable params: 11,430
Non-trainable params: 0
_________________________________________________________________
```

In [31]:

```
model_2 = model.fit(X_train, y_train, batch_size=16, validation_data=(X_test, y_test), epochs=30)
score = model.evaluate(X_test, y_test)
print("Accuracy: %.2f%%" % (score[1]*100))
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 59s 8ms/step - loss: 1.2611 - acc: 0.4713 - val_loss:
1.1306 - val_acc: 0.5490
Epoch 2/30
7352/7352 [==============================] - 57s 8ms/step - loss: 1.0592 - acc: 0.5502 - val_loss:
0.9163 - val_acc: 0.6189
Epoch 3/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.8463 - acc: 0.6113 - val_loss:
0.8728 - val_acc: 0.6223
Epoch 4/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.7502 - acc: 0.6439 - val_loss:
0.7714 - val_acc: 0.6128
Epoch 5/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.6987 - acc: 0.6676 - val_loss:
0.8003 - val_acc: 0.6464
Epoch 6/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.5929 - acc: 0.7444 - val_loss:
0.7015 - val_acc: 0.7431
Epoch 7/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.4930 - acc: 0.8368 - val_loss:
0.6798 - val_acc: 0.7710
Epoch 8/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.3855 - acc: 0.8776 - val_loss:
```

```
7352/7352 [==============================] - 57s 8ms/step - loss: [illegible] - acc: [illegible] - val_loss:
0.5434 - val_acc: 0.8426
Epoch 9/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.3314 - acc: 0.9025 - val_loss:
0.5194 - val_acc: 0.8599
Epoch 10/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.2692 - acc: 0.9163 - val_loss:
0.3865 - val_acc: 0.8768
Epoch 11/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.2740 - acc: 0.9184 - val_loss:
0.4617 - val_acc: 0.8700
Epoch 12/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.2257 - acc: 0.9289 - val_loss:
0.5521 - val_acc: 0.8514
Epoch 13/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.2152 - acc: 0.9321 - val_loss:
0.3439 - val_acc: 0.8870
Epoch 14/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.2082 - acc: 0.9343 - val_loss:
0.3788 - val_acc: 0.8890
Epoch 15/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.2099 - acc: 0.9363 - val_loss:
0.3987 - val_acc: 0.8877
Epoch 16/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.2042 - acc: 0.9368 - val_loss:
0.4684 - val_acc: 0.8707
Epoch 17/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.2096 - acc: 0.9340 - val_loss:
0.3973 - val_acc: 0.8907
Epoch 18/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.1960 - acc: 0.9343 - val_loss:
0.3536 - val_acc: 0.8819
Epoch 19/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.1718 - acc: 0.9421 - val_loss:
0.2815 - val_acc: 0.8999
Epoch 20/30
7352/7352 [==============================] - 58s 8ms/step - loss: 0.1729 - acc: 0.9463 - val_loss:
0.3410 - val_acc: 0.8907
Epoch 21/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.1576 - acc: 0.9434 - val_loss:
0.3819 - val_acc: 0.9033
Epoch 22/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.1553 - acc: 0.9461 - val_loss:
0.5065 - val_acc: 0.8497
Epoch 23/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.1378 - acc: 0.9517 - val_loss:
0.3457 - val_acc: 0.8999
Epoch 24/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.1749 - acc: 0.9416 - val_loss:
0.4547 - val_acc: 0.8867
Epoch 25/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.1641 - acc: 0.9461 - val_loss:
0.3461 - val_acc: 0.9009
Epoch 26/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.2143 - acc: 0.9372 - val_loss:
0.7472 - val_acc: 0.8616
Epoch 27/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.1950 - acc: 0.9415 - val_loss:
0.3213 - val_acc: 0.9118
Epoch 28/30
7352/7352 [==============================] - 56s 8ms/step - loss: 0.1630 - acc: 0.9475 - val_loss:
0.5020 - val_acc: 0.8918
Epoch 29/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.1494 - acc: 0.9487 - val_loss:
0.4485 - val_acc: 0.8853
Epoch 30/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.1534 - acc: 0.9474 - val_loss:
0.3223 - val_acc: 0.9057
2947/2947 [==============================] - 2s 591us/step
Accuracy: 90.57%
```

In [32]:

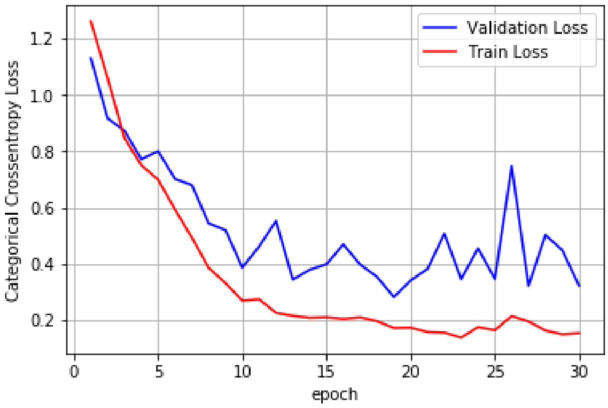
```
print(confusion_matrix(y_test, model.predict(X_test)))
```

```
Pred                  LAYING  SITTING  STANDING  WALKING  WALKING DOWNSTAIRS  \
```

```
True
LAYING                  537    0    0    0      0
SITTING                   0  406   85    0      0
STANDING                  0  109  422    1      0
WALKING                   0    0    0  469     21
WALKING_DOWNSTAIRS        0    1    0    1    415
WALKING_UPSTAIRS          1    2    0   10     38

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            0
STANDING                           0
WALKING                            6
WALKING_DOWNSTAIRS                 3
WALKING_UPSTAIRS                 420
```

In [35]:

```
fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch')
ax.set_ylabel('Categorical Crossentropy Loss')

x = list(range(1,31))

vy = model_2.history['val_loss']
ty = model_2.history['loss']
plt_dynamic(x, vy, ty, ax)
```



3. LSTM with one layer(64 units)

In [41]:

```
model = Sequential()
model.add(LSTM(64, input_shape=(timesteps, input_dim)))
model.add(Dropout(0.5))
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()

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

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
lstm_6 (LSTM)                (None, 64)                18944     
_________________________________________________________________
dropout_6 (Dropout)          (None, 64)                0         
_________________________________________________________________
dense_6 (Dense)              (None, 6)                 390       
=================================================================
Total params: 19,334
Trainable params: 19,334
Non-trainable params: 0
_________________________________________________________________
```

In [39]:

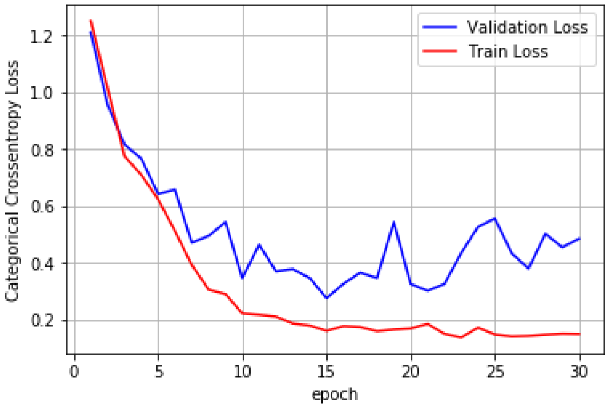
```
model_3 = model.fit(X_train, y_train, batch_size=16, validation_data=(X_test, y_test), epochs=30)
score = model.evaluate(X_test, y_test)
print("Accuracy: %.2f%%" % (score[1]*100))
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 62s 8ms/step - loss: 1.2523 - acc: 0.4600 - val_loss:
1.2109 - val_acc: 0.4744
Epoch 2/30
7352/7352 [==============================] - 60s 8ms/step - loss: 1.0148 - acc: 0.5588 - val_loss:
0.9561 - val_acc: 0.5813
Epoch 3/30
7352/7352 [==============================] - 59s 8ms/step - loss: 0.7756 - acc: 0.6483 - val_loss:
0.8169 - val_acc: 0.6403
Epoch 4/30
7352/7352 [==============================] - 60s 8ms/step - loss: 0.7094 - acc: 0.6972 - val_loss:
0.7666 - val_acc: 0.7082
Epoch 5/30
7352/7352 [==============================] - 60s 8ms/step - loss: 0.6231 - acc: 0.7470 - val_loss:
0.6421 - val_acc: 0.7574
Epoch 6/30
7352/7352 [==============================] - 60s 8ms/step - loss: 0.5116 - acc: 0.8168 - val_loss:
0.6577 - val_acc: 0.7950
Epoch 7/30
7352/7352 [==============================] - 59s 8ms/step - loss: 0.3916 - acc: 0.8724 - val_loss:
0.4703 - val_acc: 0.8548
Epoch 8/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.3047 - acc: 0.9036 - val_loss:
0.4941 - val_acc: 0.8690
Epoch 9/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2891 - acc: 0.9131 - val_loss:
0.5430 - val_acc: 0.8734
Epoch 10/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2208 - acc: 0.9264 - val_loss:
0.3454 - val_acc: 0.8931
Epoch 11/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2159 - acc: 0.9290 - val_loss:
0.4634 - val_acc: 0.8521
Epoch 12/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.2095 - acc: 0.9354 - val_loss:
0.3694 - val_acc: 0.8833
Epoch 13/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1853 - acc: 0.9370 - val_loss:
0.3773 - val_acc: 0.8921
Epoch 14/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1777 - acc: 0.9388 - val_loss:
0.3456 - val_acc: 0.8897
Epoch 15/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1598 - acc: 0.9433 - val_loss:
0.2746 - val_acc: 0.9199
Epoch 16/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1756 - acc: 0.9426 - val_loss:
0.3252 - val_acc: 0.9043
Epoch 17/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1727 - acc: 0.9437 - val_loss:
0.3647 - val_acc: 0.8948
Epoch 18/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1584 - acc: 0.9457 - val_loss:
0.3456 - val_acc: 0.8975
Epoch 19/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1642 - acc: 0.9416 - val_loss:
0.5429 - val_acc: 0.8850
Epoch 20/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1678 - acc: 0.9385 - val_loss:
0.3247 - val_acc: 0.9145
Epoch 21/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1836 - acc: 0.9354 - val_loss:
0.3013 - val_acc: 0.8955
Epoch 22/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1488 - acc: 0.9449 - val_loss:
0.3244 - val_acc: 0.9074
Epoch 23/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1360 - acc: 0.9505 - val_loss:
```

```
0.4339 - val_acc: 0.8921
Epoch 24/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1707 - acc: 0.9465 - val_loss:
0.5260 - val_acc: 0.9033
Epoch 25/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1464 - acc: 0.9461 - val_loss:
0.5555 - val_acc: 0.8833
Epoch 26/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1398 - acc: 0.9518 - val_loss:
0.4331 - val_acc: 0.9145
Epoch 27/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1415 - acc: 0.9505 - val_loss:
0.3789 - val_acc: 0.9114
Epoch 28/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1458 - acc: 0.9478 - val_loss:
0.5016 - val_acc: 0.9046
Epoch 29/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1491 - acc: 0.9495 - val_loss:
0.4541 - val_acc: 0.9030
Epoch 30/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1482 - acc: 0.9502 - val_loss:
0.4840 - val_acc: 0.9046
2947/2947 [==============================] - 2s 681us/step
Accuracy: 90.46%
```

In [44]:

```
fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch')
ax.set_ylabel('Categorical Crossentropy Loss')

x = list(range(1,31))

vy = model_3.history['val_loss']
ty = model_3.history['loss']
plt_dynamic(x, vy, ty, ax)
```



4. LSTM with one layer(32 units and 0.6 droupout)

In [45]:

```
model = Sequential()
model.add(LSTM(32, input_shape=(timesteps, input_dim)))
model.add(Dropout(0.6))
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()

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

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_7 (LSTM)                (None, 32)                5376
_________________________________________________________________
```

```
dropout_7 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_7 (Dense)              (None, 6)                 198
=================================================================
Total params: 5,574
Trainable params: 5,574
Non-trainable params: 0
_________________________________________________________________
```

In [46]:

```
model_4 = model.fit(X_train, y_train, batch_size=16, validation_data=(X_test, y_test), epochs=30)
score = model.evaluate(X_test, y_test)
print("Accuracy: %.2f%%" % (score[1]*100))
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 60s 8ms/step - loss: 1.4050 - acc: 0.3878 - val_loss:
1.2862 - val_acc: 0.4140
Epoch 2/30
7352/7352 [==============================] - 57s 8ms/step - loss: 1.1704 - acc: 0.4759 - val_loss:
1.1453 - val_acc: 0.4591
Epoch 3/30
7352/7352 [==============================] - 57s 8ms/step - loss: 1.0419 - acc: 0.5201 - val_loss:
1.0457 - val_acc: 0.4961
Epoch 4/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.8817 - acc: 0.6011 - val_loss:
0.7875 - val_acc: 0.6176
Epoch 5/30
7352/7352 [==============================] - 58s 8ms/step - loss: 0.8254 - acc: 0.6283 - val_loss:
0.9886 - val_acc: 0.5433
Epoch 6/30
7352/7352 [==============================] - 58s 8ms/step - loss: 0.8633 - acc: 0.6232 - val_loss:
0.7438 - val_acc: 0.6379
Epoch 7/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.7154 - acc: 0.6741 - val_loss:
0.7224 - val_acc: 0.6318
Epoch 8/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.6261 - acc: 0.7237 - val_loss:
0.6585 - val_acc: 0.7353
Epoch 9/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.5924 - acc: 0.7719 - val_loss:
0.5196 - val_acc: 0.7961
Epoch 10/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.5209 - acc: 0.8220 - val_loss:
0.5511 - val_acc: 0.8110
Epoch 11/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.5465 - acc: 0.8213 - val_loss:
0.4648 - val_acc: 0.8402
Epoch 12/30
7352/7352 [==============================] - 58s 8ms/step - loss: 0.4201 - acc: 0.8713 - val_loss:
0.5084 - val_acc: 0.8317
Epoch 13/30
7352/7352 [==============================] - 56s 8ms/step - loss: 0.5360 - acc: 0.8464 - val_loss:
0.8496 - val_acc: 0.7618
Epoch 14/30
7352/7352 [==============================] - 55s 7ms/step - loss: 0.3921 - acc: 0.8867 - val_loss:
0.6331 - val_acc: 0.8045
Epoch 15/30
7352/7352 [==============================] - 55s 8ms/step - loss: 0.3751 - acc: 0.8945 - val_loss:
0.6405 - val_acc: 0.8351
Epoch 16/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.3994 - acc: 0.8909 - val_loss:
0.4181 - val_acc: 0.8789
Epoch 17/30
7352/7352 [==============================] - 55s 7ms/step - loss: 0.4123 - acc: 0.8866 - val_loss:
0.3779 - val_acc: 0.8819
Epoch 18/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.3222 - acc: 0.9030 - val_loss:
0.4397 - val_acc: 0.8707
Epoch 19/30
7352/7352 [==============================] - 54s 7ms/step - loss: 0.2663 - acc: 0.9178 - val_loss:
0.4462 - val_acc: 0.8707
Epoch 20/30
7352/7352 [==============================] - 58s 8ms/step - loss: 0.2689 - acc: 0.9199 - val_loss:
```

```
0.3726 - val_acc: 0.8931
Epoch 21/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.2459 - acc: 0.9238 - val_loss:
0.4421 - val_acc: 0.8850
Epoch 22/30
7352/7352 [==============================] - 57s 8ms/step - loss: 0.2881 - acc: 0.9174 - val_loss:
0.3976 - val_acc: 0.8897
Epoch 23/30
7352/7352 [==============================] - 58s 8ms/step - loss: 0.2541 - acc: 0.9248 - val_loss:
0.9038 - val_acc: 0.7869
Epoch 24/30
7352/7352 [==============================] - 58s 8ms/step - loss: 0.2296 - acc: 0.9279 - val_loss:
0.4055 - val_acc: 0.8907
Epoch 25/30
7352/7352 [==============================] - 56s 8ms/step - loss: 0.2111 - acc: 0.9324 - val_loss:
0.5011 - val_acc: 0.8846
Epoch 26/30
7352/7352 [==============================] - 56s 8ms/step - loss: 0.2045 - acc: 0.9339 - val_loss:
0.4659 - val_acc: 0.8914
Epoch 27/30
7352/7352 [==============================] - 55s 8ms/step - loss: 0.2122 - acc: 0.9320 - val_loss:
0.5602 - val_acc: 0.8795
Epoch 28/30
7352/7352 [==============================] - 56s 8ms/step - loss: 0.1942 - acc: 0.9374 - val_loss:
0.5331 - val_acc: 0.8938
Epoch 29/30
7352/7352 [==============================] - 56s 8ms/step - loss: 0.1935 - acc: 0.9363 - val_loss:
0.3619 - val_acc: 0.9063
Epoch 30/30
7352/7352 [==============================] - 56s 8ms/step - loss: 0.1881 - acc: 0.9361 - val_loss:
0.5529 - val_acc: 0.8904
2947/2947 [==============================] - 2s 534us/step
Accuracy: 89.04%
```

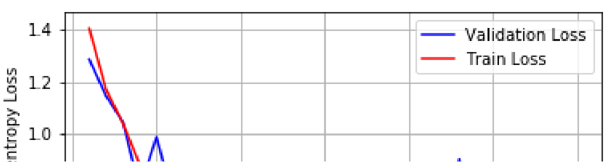
In [47]:

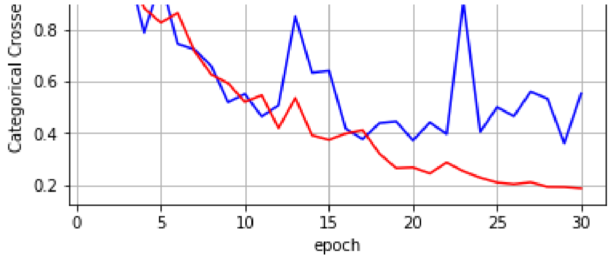
```python
print(confusion_matrix(y_test, model.predict(X_test)))
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 537        0         0        0                   0
SITTING                  1      370       107        4                   2
STANDING                 0       60       454       13                   3
WALKING                  0        0         0      485                   8
WALKING_DOWNSTAIRS       0        0         0       57                 353
WALKING_UPSTAIRS         0        1         0       27                  18

Pred                WALKING_UPSTAIRS
True
LAYING                             0
SITTING                            7
STANDING                           2
WALKING                            3
WALKING_DOWNSTAIRS                10
WALKING_UPSTAIRS                 425
```

In [48]:

```python
fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch')
ax.set_ylabel('Categorical Crossentropy Loss')

x = list(range(1,31))

vy = model_4.history['val_loss']
ty = model_4.history['loss']
plt_dynamic(x, vy, ty, ax)
```





5. LSTM with one layer(64 units and 0.6 droupout)

In [49]:

```
model = Sequential()
model.add(LSTM(64, input_shape=(timesteps, input_dim)))
model.add(Dropout(0.6))
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()

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

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
lstm_8 (LSTM)                (None, 64)                18944     
_________________________________________________________________
dropout_8 (Dropout)          (None, 64)                0         
_________________________________________________________________
dense_8 (Dense)              (None, 6)                 390       
=================================================================
Total params: 19,334
Trainable params: 19,334
Non-trainable params: 0
_________________________________________________________________
```

In [50]:

```
model_5 = model.fit(X_train, y_train, batch_size=16, validation_data=(X_test, y_test), epochs=30)
score = model.evaluate(X_test, y_test)
print("Accuracy: %.2f%%" % (score[1]*100))
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 63s 9ms/step - loss: 1.2983 - acc: 0.4444 - val_loss: 
1.1276 - val_acc: 0.5025
Epoch 2/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.9231 - acc: 0.6019 - val_loss: 
0.9085 - val_acc: 0.6071
Epoch 3/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.7759 - acc: 0.6617 - val_loss: 
0.7794 - val_acc: 0.7106
Epoch 4/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.6972 - acc: 0.7248 - val_loss: 
0.6667 - val_acc: 0.7679
Epoch 5/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.5776 - acc: 0.8011 - val_loss: 
0.5448 - val_acc: 0.8219
Epoch 6/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.4214 - acc: 0.8747 - val_loss: 
0.6507 - val_acc: 0.8191
Epoch 7/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.3039 - acc: 0.9097 - val_loss: 
0.5006 - val_acc: 0.8609
Epoch 8/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.2588 - acc: 0.9233 - val_loss: 
0.4290 - val_acc: 0.8768
Epoch 9/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.2272 - acc: 0.9289 - val_loss: 
0.4851 - val_acc: 0.8789
Epoch 10/30
```

```
Epoch 10/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.2032 - acc: 0.9319 - val_loss:
0.3241 - val_acc: 0.9009
Epoch 11/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.2932 - acc: 0.9178 - val_loss:
0.4591 - val_acc: 0.8714
Epoch 12/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.2023 - acc: 0.9313 - val_loss:
0.4719 - val_acc: 0.8941
Epoch 13/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.1783 - acc: 0.9368 - val_loss:
0.9839 - val_acc: 0.8466
Epoch 14/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1944 - acc: 0.9396 - val_loss:
0.4976 - val_acc: 0.8965
Epoch 15/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1776 - acc: 0.9421 - val_loss:
0.5102 - val_acc: 0.8999
Epoch 16/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1928 - acc: 0.9410 - val_loss:
0.4409 - val_acc: 0.8856
Epoch 17/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.1559 - acc: 0.9444 - val_loss:
0.3759 - val_acc: 0.9060
Epoch 18/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.1742 - acc: 0.9455 - val_loss:
0.3577 - val_acc: 0.9030
Epoch 19/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.1674 - acc: 0.9438 - val_loss:
0.4104 - val_acc: 0.8972
Epoch 20/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.1695 - acc: 0.9399 - val_loss:
0.3907 - val_acc: 0.9080
Epoch 21/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1469 - acc: 0.9459 - val_loss:
0.4483 - val_acc: 0.9030
Epoch 22/30
7352/7352 [==============================] - 63s 9ms/step - loss: 0.1489 - acc: 0.9450 - val_loss:
0.3794 - val_acc: 0.9104
Epoch 23/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.1412 - acc: 0.9456 - val_loss:
0.3606 - val_acc: 0.9148
Epoch 24/30
7352/7352 [==============================] - 64s 9ms/step - loss: 0.2848 - acc: 0.9155 - val_loss:
0.4684 - val_acc: 0.8616
Epoch 25/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1836 - acc: 0.9376 - val_loss:
0.5604 - val_acc: 0.9016
Epoch 26/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1727 - acc: 0.9407 - val_loss:
0.3755 - val_acc: 0.8992
Epoch 27/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.1445 - acc: 0.9501 - val_loss:
0.5063 - val_acc: 0.9009
Epoch 28/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.1427 - acc: 0.9467 - val_loss:
0.4127 - val_acc: 0.9131
Epoch 29/30
7352/7352 [==============================] - 61s 8ms/step - loss: 0.1573 - acc: 0.9467 - val_loss:
0.6957 - val_acc: 0.8968
Epoch 30/30
7352/7352 [==============================] - 62s 8ms/step - loss: 0.1747 - acc: 0.9461 - val_loss:
0.5408 - val_acc: 0.8979
2947/2947 [==============================] - 2s 632us/step
Accuracy: 89.79%
```

In [51]:

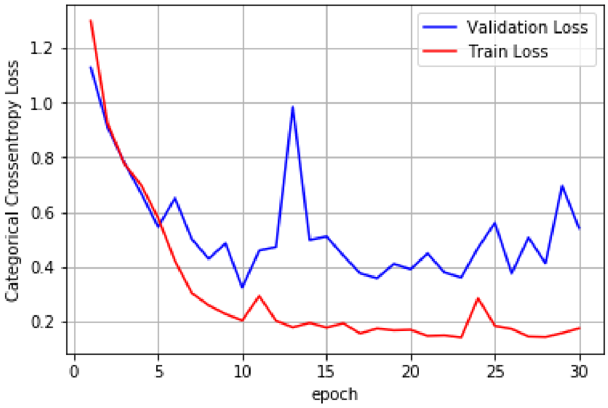
```python
print(confusion_matrix(y_test, model.predict(X_test)))
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 513        3         0        0                   0
SITTING                  0      416        72        0                   0
STANDING                 0      114       415        3                   0
WALKING                  0        0         1      469                  25
```

```
WALKING_DOWNSTAIRS        0         0          0        11                   409
WALKING_UPSTAIRS          0         3          1        27                    16

Pred                WALKING_UPSTAIRS
True
LAYING                            21
SITTING                            3
STANDING                           0
WALKING                            1
WALKING_DOWNSTAIRS                 0
WALKING_UPSTAIRS                 424
```

In [52]:

```
fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch')
ax.set_ylabel('Categorical Crossentropy Loss')

x = list(range(1,31))

vy = model_5.history['val_loss']
ty = model_5.history['loss']
plt_dynamic(x, vy, ty, ax)
```



6. LSTM with two layers(0.75 droupout)

In [56]:

```
model = Sequential()
#Layer_1
model.add(LSTM(64, return_sequences=True, input_shape=(timesteps, input_dim)))
model.add(Dropout(0.75))
#Layer_2
model.add(LSTM(32))
model.add(Dropout(0.75))
model.add(Dense(n_classes, activation='sigmoid'))
model.summary()

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

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_15 (LSTM)               (None, 128, 64)           18944
_________________________________________________________________
dropout_12 (Dropout)         (None, 128, 64)           0
_________________________________________________________________
lstm_16 (LSTM)               (None, 32)                12416
_________________________________________________________________
dropout_13 (Dropout)         (None, 32)                0
_________________________________________________________________
dense_9 (Dense)              (None, 6)                 198
=================================================================
Total params: 31,558
```

```
Trainable params: 31,558
Non-trainable params: 0
_________________________________________________________________
```

In [58]:

```python
model_6 = model.fit(X_train, y_train, batch_size=128, validation_data=(X_test, y_test), epochs=30)
score = model.evaluate(X_test, y_test)
print("Accuracy: %.2f%%" % (score[1]*100))
```

```
Train on 7352 samples, validate on 2947 samples
Epoch 1/30
7352/7352 [==============================] - 24s 3ms/step - loss: 1.2402 - acc: 0.5011 - val_loss:
1.1058 - val_acc: 0.5677
Epoch 2/30
7352/7352 [==============================] - 24s 3ms/step - loss: 1.1066 - acc: 0.5431 - val_loss:
0.9819 - val_acc: 0.6549
Epoch 3/30
7352/7352 [==============================] - 24s 3ms/step - loss: 1.0185 - acc: 0.5698 - val_loss:
0.8902 - val_acc: 0.6702
Epoch 4/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.9626 - acc: 0.5705 - val_loss:
0.8544 - val_acc: 0.6254
Epoch 5/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.9081 - acc: 0.5958 - val_loss:
0.8031 - val_acc: 0.6512
Epoch 6/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.8665 - acc: 0.6062 - val_loss:
0.7871 - val_acc: 0.6882
Epoch 7/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.8739 - acc: 0.6186 - val_loss:
1.2496 - val_acc: 0.5314
Epoch 8/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.8424 - acc: 0.6427 - val_loss:
0.7753 - val_acc: 0.6335
Epoch 9/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.7965 - acc: 0.6416 - val_loss:
0.7289 - val_acc: 0.7021
Epoch 10/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.7798 - acc: 0.6651 - val_loss:
0.7115 - val_acc: 0.6885
Epoch 11/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.7733 - acc: 0.6586 - val_loss:
0.7066 - val_acc: 0.6966
Epoch 12/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.7723 - acc: 0.6661 - val_loss:
0.6873 - val_acc: 0.7000
Epoch 13/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.7368 - acc: 0.6704 - val_loss:
0.6892 - val_acc: 0.7099
Epoch 14/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.7369 - acc: 0.6780 - val_loss:
0.6658 - val_acc: 0.6159
Epoch 15/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.7234 - acc: 0.6661 - val_loss:
0.6644 - val_acc: 0.6230
Epoch 16/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.7051 - acc: 0.6685 - val_loss:
0.6654 - val_acc: 0.6278
Epoch 17/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.6971 - acc: 0.6778 - val_loss:
0.7794 - val_acc: 0.6271
Epoch 18/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.6794 - acc: 0.6919 - val_loss:
0.6279 - val_acc: 0.6328
Epoch 19/30
7352/7352 [==============================] - 23s 3ms/step - loss: 0.6635 - acc: 0.7031 - val_loss:
0.5681 - val_acc: 0.6973
Epoch 20/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.6429 - acc: 0.7174 - val_loss:
0.6050 - val_acc: 0.7374
Epoch 21/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.6588 - acc: 0.7040 - val_loss:
0.6940 - val_acc: 0.6929
Epoch 22/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.6459 - acc: 0.7097 - val_loss:
```

```
0.6723 - val_acc: 0.6895
Epoch 23/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.6204 - acc: 0.7239 - val_loss:
0.7893 - val_acc: 0.7475
Epoch 24/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.5994 - acc: 0.7542 - val_loss:
0.5776 - val_acc: 0.8347
Epoch 25/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.6136 - acc: 0.7462 - val_loss:
0.6281 - val_acc: 0.7903
Epoch 26/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.5885 - acc: 0.7631 - val_loss:
0.5293 - val_acc: 0.8127
Epoch 27/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.6172 - acc: 0.7787 - val_loss:
0.5413 - val_acc: 0.8622
Epoch 28/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.5593 - acc: 0.8003 - val_loss:
0.5530 - val_acc: 0.8544
Epoch 29/30
7352/7352 [==============================] - 24s 3ms/step - loss: 0.5302 - acc: 0.8108 - val_loss:
0.5564 - val_acc: 0.8473
Epoch 30/30
7352/7352 [==============================] - 25s 3ms/step - loss: 0.5401 - acc: 0.8290 - val_loss:
0.4915 - val_acc: 0.8738
2947/2947 [==============================] - 4s 1ms/step
Accuracy: 87.38%
```

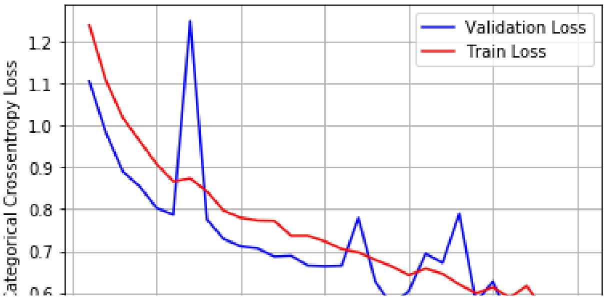
In [59]:

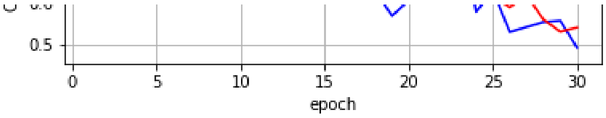
```python
print(confusion_matrix(y_test, model.predict(X_test)))
```

```
Pred                LAYING  SITTING  STANDING  WALKING  WALKING_DOWNSTAIRS  \
True
LAYING                 510        0         0        0                   0
SITTING                  6      382        99        4                   0
STANDING                 0       78       453        1                   0
WALKING                  0        0         0      467                  16
WALKING_DOWNSTAIRS       0        0         0       15                 370
WALKING_UPSTAIRS         0        0         0       35                  43

Pred                WALKING_UPSTAIRS
True
LAYING                            27
SITTING                            0
STANDING                           0
WALKING                           13
WALKING_DOWNSTAIRS                35
WALKING_UPSTAIRS                 393
```

In [60]:

```python
fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch')
ax.set_ylabel('Categorical Crossentropy Loss')

x = list(range(1,31))

vy = model_6.history['val_loss']
ty = model_6.history['loss']
plt_dynamic(x, vy, ty, ax)
```





Conclusion

1. Initially, we had two types of data. One with the expert engineering features and the other one is raw data.
2. We've taken the raw data to apply Deep Learning models like LSTM, which is a type of RNN, on top of it.
3. We've converted the raw data into 128 dimenssions vector from the 9 time series raw data of accelerometer and gyroscope readings.
4. Later, we've tried various architechtures with different units, dropouts and layers of the LSTM networks on the Raw data.
5. Even though we don't have a huge data, with the limited data we had, we got to see a best accuracy of 91.48% which is pretty good.

```
In [ ]:
```