XGB&NN

October 31, 2019

[1]:
```
import os,sys
import pandas as pd
from sklearn.model_selection import train_test_split
import scipy.io
import numpy as np
from scipy.spatial.distance import pdist
import time
import math
import xgboost as xgb
from xgboost.sklearn import XGBClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import GridSearchCV
import keras
from keras.utils import to_categorical
import matplotlib.pyplot as plt
from tensorflow.keras.models import Sequential
```

```
Using TensorFlow backend.
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorflow/python/framework/dtypes.py:516: FutureWarning: Passing
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorflow/python/framework/dtypes.py:517: FutureWarning: Passing
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorflow/python/framework/dtypes.py:518: FutureWarning: Passing
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorflow/python/framework/dtypes.py:519: FutureWarning: Passing
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
```

```
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorflow/python/framework/dtypes.py:520: FutureWarning: Passing
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorflow/python/framework/dtypes.py:525: FutureWarning: Passing
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:541: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:542: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:543: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:544: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:545: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:550: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

[2]:
```
from keras.layers import Dense, Activation, Flatten, Input, Dropout
from keras.layers import BatchNormalization
from keras.models import Model
from keras import initializers
from keras.optimizers import Adam
```

[3]:
```
# change the root to your own path
root = sys.path[0]
train_dir =  os.path.join(root,  '../data/train_set')
train_image_dir =  os.path.join(train_dir, 'images')
train_pt_dir =  os.path.join(train_dir, 'points' )
train_label_path =  os.path.join(train_dir,  "label.csv")
```

[4]:
```
info = pd.read_csv(train_label_path)
train_idx, test_idx = train_test_split(info['Index'], test_size=0.2,␣
 ↪random_state=123)

# read mat file and store coordinates in mat
m = []
for idx in info['Index']:
    file = "%04d.mat"%(idx)
    m.append( scipy.io.loadmat( os.path.join( train_pt_dir, file ) ))

mat = [x[[i for i in x.keys() if not i in ['__header__', '__version__',␣
 ↪'__globals__']][0]] for x in m]
```

0.0.1 train test split

[5]:
```
train_idx, test_idx = train_test_split(info['Index'], test_size=0.2,␣
 ↪random_state=123)
```

[16]:
```
train_mat = [ mat[i-1] for i in train_idx ]
test_mat = [ mat[i-1] for i in test_idx ]
```

[22]:
```
train_labels = info.emotion_idx[train_idx-1]
test_labels = info.emotion_idx[test_idx-1]
```

[29]:
```
train_label_cat = to_categorical(train_labels)
train_label_cat= train_label_cat[:,1:]
test_label_cat = to_categorical(test_labels)
test_label_cat= test_label_cat[:,1:]
```

0.0.2 feature extraction

[18]:
```
# method 1  pairwise_dist_cal
#def pairwise_dist_cal(xy_cord):
    #p_dist =[]
    #for i in range(xy_cord.shape[0]):
        #for j in range(i+1,xy_cord.shape[0]):
                # p_dist.append( abs(round(xy_cord[i,0]) - round(xy_cord[j,0] ))␣
 ↪)
                # p_dist.append( abs(round(xy_cord[i,1]) - round(xy_cord[j,1] )␣
 ↪) )
    #return p_dist



#### updated methods with selected fiducial points; this reduce 78 poins to 50
'''
feature selection : 1. remove  points P64 - 70 and P72 - 78
                    2. remove P51,53,55,57,58,60, 61, 63
                    3. calculate midpints between  upper and lower eyebrow lines␣
 ↪and replace P20-22,P24-16 with midpoints
So there is in total 50 points left, which gives 50*49 pairwise distance
'''

def pairwise_dist_cal_updt(mt):
    t0 = time.time()
    p_dist_updt =np.zeros([len(mt),1225,2])
    n = len(mt)
    for k in range(n):
        xy_cord = mt[k]

        xy_cord_cpy  = xy_cord

        #  eye_brow midpoint
        to_add_brl = (xy_cord_cpy[19:22]+ xy_cord_cpy[23:26])/2
        to_add_brr = (xy_cord_cpy[27:30]+ xy_cord_cpy[31:34])/2

        # index to remove
        rm_idx = np.append(np.arange(63,70), np.arange(71,78))
        rm_idx = np.append(rm_idx,np.arange(50,57,2) )
        rm_idx = np.append(rm_idx, [57,59,60,62])
        rm_idx = np.append(rm_idx, np.arange(19,22))
        rm_idx = np.append(rm_idx, np.arange(23,26))
        rm_idx = np.append(rm_idx, np.arange(27,30))
        rm_idx = np.append(rm_idx, np.arange(31,34))
        xy_cord = np.delete(xy_cord, rm_idx, axis = 0)
        xy_cord = np.concatenate((xy_cord,to_add_brl,to_add_brr))


        dist_h = []
        dist_v = []
        for i in range(xy_cord.shape[0]):
            for j in range(i+1,xy_cord.shape[0]):
                dist_h.append( abs(round(xy_cord[i,0]) - round(xy_cord[j,0] )) )
                dist_v.append( abs(round(xy_cord[i,1]) - round(xy_cord[j,1] ) )␣
↪)
        p_dist_updt[k,:,0]= dist_h
        p_dist_updt[k,:,1]= dist_v


    print("feature constructions takes %s seconds" % (time.time() - t0))
    return p_dist_updt.reshape([n,2450])
```

[19]:
```
print("training: ")
train_data = pairwise_dist_cal_updt(train_mat[0:])
```

```
training:
training and testing feature constructions 18.035425901412964 seconds
```

[21]:
```
print("testing: ")
test_data = pairwise_dist_cal_updt(test_mat[0:])
```

```
testing:
training and testing feature constructions 4.777792930603027 seconds
```

[ ]:
```
accuracy_score(pred_xgb_2,test_labels_xgb )
```

1 XGB model

[55]:
```
from xgboost.sklearn import XGBClassifier
```

[58]:
```
train_labels_xgb = [ x  - 1 for x in train_labels ]
test_labels_xgb = [ x  - 1 for x in test_labels ]
```

[60]:
```
start_time=time.time()

xgb = XGBClassifier(
 learning_rate =0.1,
 n_estimators= 200,
 max_depth=5,
 min_child_weight=1,
 gamma=0,
 subsample=0.8,
 colsample_bytree=0.8,
 objective= 'multi:softmax',  # for multi-labels classification
 num_class = 22,
 scale_pos_weight=1,
 seed=123)
```

```
xgb.fit(train_data, train_labels_xgb ,eval_metric='auc')
print("training  model takes %s seconds" % round((time.time() - start_time),3))
```

```
training  model takes 1074.819 seconds
```

```
[65]: start_time = time.time()
pred_xgb = xgb.predict(test_data)
print("testing model takes %s seconds" % round((time.time() - start_time),3))
```

```
testing model takes 0.71 seconds
```

```
[68]: acc_xgb = accuracy_score(pred_xgb,test_labels_xgb )
print("Test accuracy is %s percent" %(acc_xgb*100))
```

```
Test accuracy is 47.4 percent
```

Try other leaner rate and n_trees

```
[96]: xgb_2 = XGBClassifier(
 learning_rate =0.01,
 n_estimators= 500,
 max_depth= 3 ,
 min_child_weight=1,
 gamma=0,
 subsample=0.8,
 colsample_bytree=0.8,
 objective= 'multi:softmax',  # for multi-labels classification
 num_class = 22,
 scale_pos_weight=1,
 seed=123)
```

```
[97]: start_time = time.time()
xgb_2.fit(train_data, train_labels_xgb ,eval_metric='auc')
print("training  model takes %s seconds" % round((time.time() - start_time),3))
```

```
[97]: XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=0.8, gamma=0,
              learning_rate=0.01, max_delta_step=0, max_depth=3,
              min_child_weight=1, missing=None, n_estimators=500, n_jobs=1,
              nthread=None, num_class=22, objective='multi:softprob',
              random_state=0, reg_alpha=0, reg_lambda=1, scale_pos_weight=1,
              seed=123, silent=None, subsample=0.8, verbosity=1)
```

```
[115]: acc_sc_tra = accuracy_score(pred_xgb_2_train,train_labels_xgb )
print("Train accuracy is %s percent" %(acc_sc_tra *100))
```

```
Test accuracy is 98.15 percent
```

[100]:
```
start_time = time.time()
pred_xgb_2 = xgb_2.predict(test_data)
print("testing model takes %s seconds" % round((time.time() - start_time),3))
```

```
testing model takes 1.546 seconds
```

[113]:
```
pred_xgb_2_train =  xgb_2.predict(train_data)
```

2 NN model

[32]:
```
X = train_data
Y = train_label_cat
```

[126]:
```
input_shape = [2450]
input_layer = Input(input_shape)
x = BatchNormalization()(input_layer)
x = Dense(22*20,activation='relu',kernel_initializer=initializers.
 ↪glorot_normal(seed=4))(x)
x = Dropout(0.25)(x)
x = BatchNormalization()(x)
x = Dense(22*10,activation='relu',kernel_initializer=initializers.
 ↪glorot_normal(seed=4))(x)
x = Dropout(0.25)(x)
x = Dense(22*6,activation='relu',kernel_initializer=initializers.
 ↪glorot_normal(seed=4))(x)
x = Dropout(0.25)(x)
x = Dense(22*2,activation='relu',kernel_initializer=initializers.
 ↪glorot_normal(seed=4))(x)
output_layer = Dense(22,activation='softmax',kernel_initializer=initializers.
 ↪glorot_normal(seed=4))(x)
model = Model(input_layer,output_layer)
```

[127]:
```
start_time = time.time()
model.compile(loss='categorical_crossentropy',optimizer = Adam(lr=0.
 ↪001),metrics=['accuracy'])
model_his = model.fit(X,Y,epochs=100)
plt.figure(figsize=[8,6])
plt.plot(model_his.history['accuracy'],'r',linewidth=1.0)
plt.legend(['Training Accuracy'],fontsize=18)
plt.xlabel('Epochs ',fontsize=16)
plt.ylabel('Accuracy',fontsize=16)
plt.title('Accuracy Curves',fontsize=16)
print("training  model takes %s seconds" % round((time.time() - start_time),3))
```

```
Epoch 1/100
```

```
2000/2000 [==============================] - 3s 1ms/step - loss: 3.0529 -
accuracy: 0.0985
Epoch 2/100
2000/2000 [==============================] - 2s 909us/step - loss: 2.6859 -
accuracy: 0.1805
Epoch 3/100
2000/2000 [==============================] - 2s 859us/step - loss: 2.3663 -
accuracy: 0.2430
Epoch 4/100
2000/2000 [==============================] - 2s 920us/step - loss: 2.1117 -
accuracy: 0.3240
Epoch 5/100
2000/2000 [==============================] - 2s 1ms/step - loss: 2.0121 -
accuracy: 0.3495
Epoch 6/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.7988 -
accuracy: 0.4005
Epoch 7/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.7288 -
accuracy: 0.4230
Epoch 8/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.6655 -
accuracy: 0.4445
Epoch 9/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.5426 -
accuracy: 0.4745
Epoch 10/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.5484 -
accuracy: 0.4765
Epoch 11/100
2000/2000 [==============================] - 2s 820us/step - loss: 1.4704 -
accuracy: 0.5025
Epoch 12/100
2000/2000 [==============================] - 2s 858us/step - loss: 1.4005 -
accuracy: 0.5345
Epoch 13/100
2000/2000 [==============================] - 2s 917us/step - loss: 1.3527 -
accuracy: 0.5360
Epoch 14/100
2000/2000 [==============================] - 2s 921us/step - loss: 1.3314 -
accuracy: 0.5450
Epoch 15/100
2000/2000 [==============================] - 2s 918us/step - loss: 1.3508 -
accuracy: 0.5320
Epoch 16/100
2000/2000 [==============================] - 2s 931us/step - loss: 1.2606 -
accuracy: 0.5740
Epoch 17/100
```

```
2000/2000 [==============================] - 2s 954us/step - loss: 1.2482 -
accuracy: 0.5765
Epoch 18/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.1849 -
accuracy: 0.5980
Epoch 19/100
2000/2000 [==============================] - 2s 840us/step - loss: 1.1754 -
accuracy: 0.6040
Epoch 20/100
2000/2000 [==============================] - 2s 941us/step - loss: 1.1311 -
accuracy: 0.6055
Epoch 21/100
2000/2000 [==============================] - 2s 817us/step - loss: 1.1277 -
accuracy: 0.6070
Epoch 22/100
2000/2000 [==============================] - 2s 769us/step - loss: 1.1193 -
accuracy: 0.6010
Epoch 23/100
2000/2000 [==============================] - 2s 836us/step - loss: 1.0977 -
accuracy: 0.6265
Epoch 24/100
2000/2000 [==============================] - 2s 824us/step - loss: 1.0538 -
accuracy: 0.6380
Epoch 25/100
2000/2000 [==============================] - 2s 881us/step - loss: 1.0279 -
accuracy: 0.6530
Epoch 26/100
2000/2000 [==============================] - 2s 817us/step - loss: 1.0385 -
accuracy: 0.6385
Epoch 27/100
2000/2000 [==============================] - 2s 798us/step - loss: 0.9449 -
accuracy: 0.6715
Epoch 28/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.9089 -
accuracy: 0.6865
Epoch 29/100
2000/2000 [==============================] - 2s 837us/step - loss: 0.9294 -
accuracy: 0.6770
Epoch 30/100
2000/2000 [==============================] - 3s 2ms/step - loss: 0.9306 -
accuracy: 0.6750
Epoch 31/100
2000/2000 [==============================] - 3s 2ms/step - loss: 0.9257 -
accuracy: 0.6815
Epoch 32/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.8786 -
accuracy: 0.6990
Epoch 33/100
```

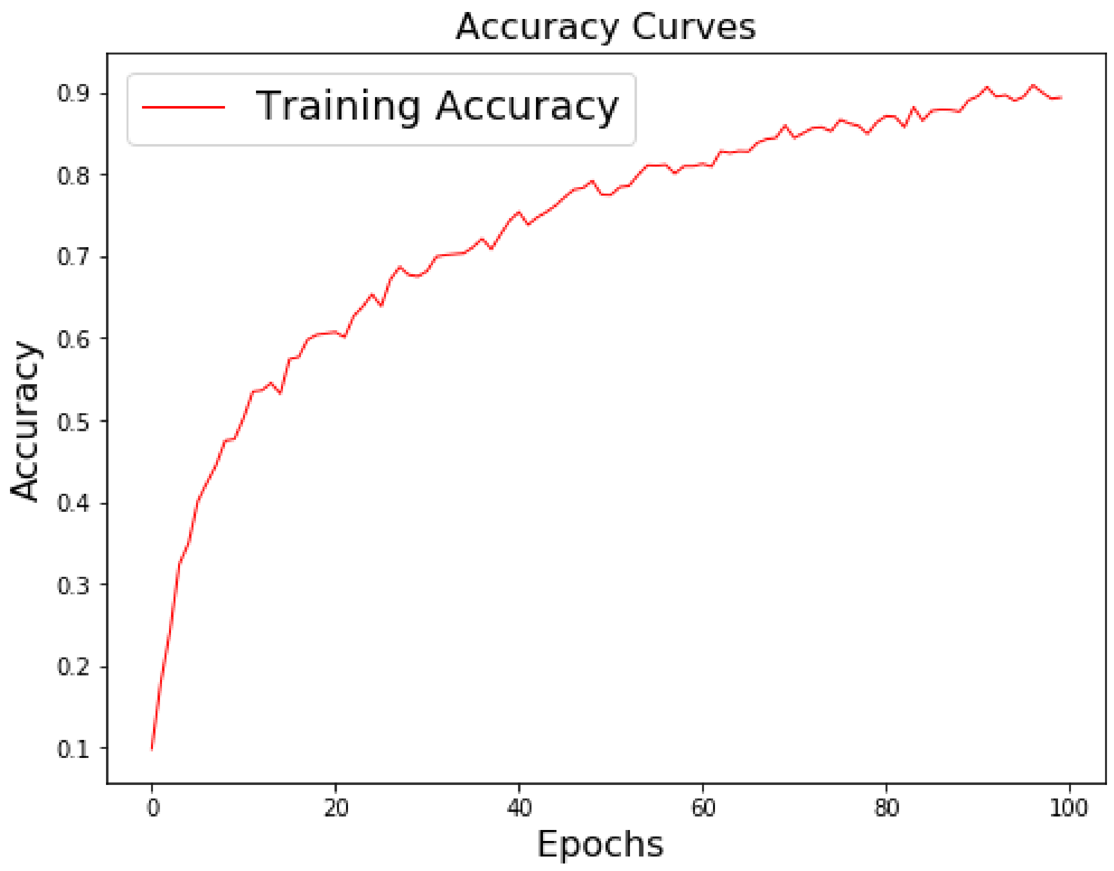
```
2000/2000 [==============================] - 2s 951us/step - loss: 0.8756 -
accuracy: 0.7010
Epoch 34/100
2000/2000 [==============================] - 1s 735us/step - loss: 0.8754 -
accuracy: 0.7020
Epoch 35/100
2000/2000 [==============================] - 2s 769us/step - loss: 0.8581 -
accuracy: 0.7030
Epoch 36/100
2000/2000 [==============================] - 1s 742us/step - loss: 0.8280 -
accuracy: 0.7105
Epoch 37/100
2000/2000 [==============================] - 1s 733us/step - loss: 0.8125 -
accuracy: 0.7210
Epoch 38/100
2000/2000 [==============================] - 1s 731us/step - loss: 0.8386 -
accuracy: 0.7080
Epoch 39/100
2000/2000 [==============================] - 1s 737us/step - loss: 0.8213 -
accuracy: 0.7265
Epoch 40/100
2000/2000 [==============================] - 1s 727us/step - loss: 0.7536 -
accuracy: 0.7430
Epoch 41/100
2000/2000 [==============================] - 1s 744us/step - loss: 0.7249 -
accuracy: 0.7535
Epoch 42/100
2000/2000 [==============================] - 1s 728us/step - loss: 0.7426 -
accuracy: 0.7380
Epoch 43/100
2000/2000 [==============================] - 2s 806us/step - loss: 0.7287 -
accuracy: 0.7470
Epoch 44/100
2000/2000 [==============================] - 2s 793us/step - loss: 0.6930 -
accuracy: 0.7535
Epoch 45/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.6562 -
accuracy: 0.7615
Epoch 46/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.6737 -
accuracy: 0.7715
Epoch 47/100
2000/2000 [==============================] - 2s 970us/step - loss: 0.6686 -
accuracy: 0.7810
Epoch 48/100
2000/2000 [==============================] - 2s 976us/step - loss: 0.6349 -
accuracy: 0.7830
Epoch 49/100
```

```
2000/2000 [==============================] - 3s 1ms/step - loss: 0.6129 -
accuracy: 0.7915
Epoch 50/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.6602 -
accuracy: 0.7745
Epoch 51/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.6630 -
accuracy: 0.7740
Epoch 52/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.6274 -
accuracy: 0.7840
Epoch 53/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.6272 -
accuracy: 0.7855
Epoch 54/100
2000/2000 [==============================] - 3s 2ms/step - loss: 0.5855 -
accuracy: 0.7990
Epoch 55/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.5588 -
accuracy: 0.8105
Epoch 56/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.5393 -
accuracy: 0.8100
Epoch 57/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.5586 -
accuracy: 0.8110
Epoch 58/100
2000/2000 [==============================] - 3s 2ms/step - loss: 0.5711 -
accuracy: 0.8005
Epoch 59/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.5400 -
accuracy: 0.8095
Epoch 60/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.5647 -
accuracy: 0.8095
Epoch 61/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.5512 -
accuracy: 0.8120
Epoch 62/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.5687 -
accuracy: 0.8090
Epoch 63/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.4959 -
accuracy: 0.8275
Epoch 64/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.4797 -
accuracy: 0.8255
Epoch 65/100
```

```
2000/2000 [==============================] - 2s 1ms/step - loss: 0.5126 -
accuracy: 0.8275
Epoch 66/100
2000/2000 [==============================] - 2s 907us/step - loss: 0.4889 -
accuracy: 0.8270
Epoch 67/100
2000/2000 [==============================] - 2s 952us/step - loss: 0.5032 -
accuracy: 0.8380
Epoch 68/100
2000/2000 [==============================] - 2s 908us/step - loss: 0.4741 -
accuracy: 0.8425
Epoch 69/100
2000/2000 [==============================] - 2s 939us/step - loss: 0.4445 -
accuracy: 0.8440
Epoch 70/100
2000/2000 [==============================] - 2s 923us/step - loss: 0.4225 -
accuracy: 0.8590
Epoch 71/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.4538 -
accuracy: 0.8440
Epoch 72/100
2000/2000 [==============================] - 2s 939us/step - loss: 0.4599 -
accuracy: 0.8500
Epoch 73/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.4199 -
accuracy: 0.8560
Epoch 74/100
2000/2000 [==============================] - 2s 1000us/step - loss: 0.3957 -
accuracy: 0.8570
Epoch 75/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.4267 -
accuracy: 0.8520
Epoch 76/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.4307 -
accuracy: 0.8660
Epoch 77/100
2000/2000 [==============================] - 3s 2ms/step - loss: 0.4199 -
accuracy: 0.8615
Epoch 78/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.4151 -
accuracy: 0.8585
Epoch 79/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.4323 -
accuracy: 0.8490
Epoch 80/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3866 -
accuracy: 0.8630
Epoch 81/100
```

```
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3815 -
accuracy: 0.8705
Epoch 82/100
2000/2000 [==============================] - 2s 999us/step - loss: 0.4028 -
accuracy: 0.8695
Epoch 83/100
2000/2000 [==============================] - 2s 957us/step - loss: 0.4245 -
accuracy: 0.8570
Epoch 84/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3733 -
accuracy: 0.8815
Epoch 85/100
2000/2000 [==============================] - 2s 927us/step - loss: 0.3980 -
accuracy: 0.8650
Epoch 86/100
2000/2000 [==============================] - 2s 911us/step - loss: 0.3910 -
accuracy: 0.8770
Epoch 87/100
2000/2000 [==============================] - 2s 849us/step - loss: 0.3542 -
accuracy: 0.8785
Epoch 88/100
2000/2000 [==============================] - 2s 835us/step - loss: 0.3880 -
accuracy: 0.8780
Epoch 89/100
2000/2000 [==============================] - 2s 897us/step - loss: 0.3690 -
accuracy: 0.8760
Epoch 90/100
2000/2000 [==============================] - 2s 955us/step - loss: 0.3297 -
accuracy: 0.8895
Epoch 91/100
2000/2000 [==============================] - 2s 828us/step - loss: 0.3331 -
accuracy: 0.8945
Epoch 92/100
2000/2000 [==============================] - 2s 898us/step - loss: 0.2733 -
accuracy: 0.9060
Epoch 93/100
2000/2000 [==============================] - 2s 859us/step - loss: 0.3003 -
accuracy: 0.8940
Epoch 94/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.3209 -
accuracy: 0.8960
Epoch 95/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3431 -
accuracy: 0.8890
Epoch 96/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3300 -
accuracy: 0.8940
Epoch 97/100
```

```
2000/2000 [==============================] - 2s 1ms/step - loss: 0.2941 -
accuracy: 0.9085
Epoch 98/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3094 -
accuracy: 0.8995
Epoch 99/100
2000/2000 [==============================] - 2s 966us/step - loss: 0.3041 -
accuracy: 0.8915
Epoch 100/100
2000/2000 [==============================] - 2s 985us/step - loss: 0.3032 -
accuracy: 0.8930
training  model takes 209.942 seconds
```



[128]:
```python
t0 = time.time()
pred = model.predict(test_data)
pred_lst = []
for i in range(len(pred)):
    arr = pred[i]
    idx = np.argwhere(arr == np.max(arr))
    pred_lst.append(idx[0][0])
```

```
tst_labl = np.argmax(test_label_cat, axis=-1)
acc = accuracy_score(pred_lst, tst_labl)
print("Test accuracy is %s percent" %(acc*100))
print("testing model takes %s seconds" % round((time.time() - t0),3))
```

```
Test accuracy is 52.400000000000006 percent
testing model takes 0.844 seconds
```

[129]:
```
input_shape = [2450]
input_layer = Input(input_shape)
x = BatchNormalization(momentum = 0.88)(input_layer)
x = Dense(22*10,activation='relu',kernel_initializer=initializers.
 ↪glorot_normal(seed=4))(x)
x = Dropout(0.25)(x)
x = BatchNormalization()(x)
x = Dense(22*8,activation='relu',kernel_initializer=initializers.
 ↪glorot_normal(seed=4))(x)
x = Dropout(0.25)(x)
x = Dense(22*4,activation='relu',kernel_initializer=initializers.
 ↪glorot_normal(seed=4))(x)
x = Dropout(0.25)(x)
x = Dense(22*2,activation='relu',kernel_initializer=initializers.
 ↪glorot_normal(seed=4))(x)
output_layer = Dense(22,activation='softmax',kernel_initializer=initializers.
 ↪glorot_normal(seed=4))(x)
model2 = Model(input_layer,output_layer)
```

[130]:
```
start_time = time.time()
model2.compile(loss='categorical_crossentropy',optimizer = Adam(lr=0.
 ↪001),metrics=['accuracy'])
model_his = model2.fit(X,Y,epochs=100)
plt.figure(figsize=[8,6])
plt.plot(model_his.history['accuracy'],'r',linewidth=1.0)
plt.legend(['Training Accuracy'],fontsize=18)
plt.xlabel('Epochs ',fontsize=16)
plt.ylabel('Accuracy',fontsize=16)
plt.title('Accuracy Curves',fontsize=16)
print("training  model takes %s seconds" % round((time.time() - start_time),3))
```

```
Epoch 1/100
2000/2000 [==============================] - 2s 979us/step - loss: 3.0595 -
accuracy: 0.0820
Epoch 2/100
2000/2000 [==============================] - 1s 477us/step - loss: 2.7241 -
accuracy: 0.1575
Epoch 3/100
2000/2000 [==============================] - 1s 499us/step - loss: 2.4446 -
```

```
accuracy: 0.2295
Epoch 4/100
2000/2000 [==============================] - 1s 676us/step - loss: 2.1716 -
accuracy: 0.3045
Epoch 5/100
2000/2000 [==============================] - 1s 551us/step - loss: 2.0197 -
accuracy: 0.3385
Epoch 6/100
2000/2000 [==============================] - 1s 572us/step - loss: 1.8954 -
accuracy: 0.3710
Epoch 7/100
2000/2000 [==============================] - 1s 600us/step - loss: 1.8180 -
accuracy: 0.3970
Epoch 8/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.7580 -
accuracy: 0.4250
Epoch 9/100
2000/2000 [==============================] - 1s 576us/step - loss: 1.6953 -
accuracy: 0.4280
Epoch 10/100
2000/2000 [==============================] - 1s 498us/step - loss: 1.5777 -
accuracy: 0.4610
Epoch 11/100
2000/2000 [==============================] - 1s 460us/step - loss: 1.5429 -
accuracy: 0.4900
Epoch 12/100
2000/2000 [==============================] - 1s 494us/step - loss: 1.4990 -
accuracy: 0.4865
Epoch 13/100
2000/2000 [==============================] - 1s 673us/step - loss: 1.5019 -
accuracy: 0.4955
Epoch 14/100
2000/2000 [==============================] - 1s 576us/step - loss: 1.4658 -
accuracy: 0.4980
Epoch 15/100
2000/2000 [==============================] - 1s 513us/step - loss: 1.4290 -
accuracy: 0.5060
Epoch 16/100
2000/2000 [==============================] - 1s 512us/step - loss: 1.4187 -
accuracy: 0.5155
Epoch 17/100
2000/2000 [==============================] - 1s 502us/step - loss: 1.3351 -
accuracy: 0.5440
Epoch 18/100
2000/2000 [==============================] - 1s 513us/step - loss: 1.3328 -
accuracy: 0.5485
Epoch 19/100
2000/2000 [==============================] - 1s 489us/step - loss: 1.2941 -
```

```
accuracy: 0.5425
Epoch 20/100
2000/2000 [==============================] - 1s 564us/step - loss: 1.2771 -
accuracy: 0.5710
Epoch 21/100
2000/2000 [==============================] - 1s 735us/step - loss: 1.2435 -
accuracy: 0.5705
Epoch 22/100
2000/2000 [==============================] - 1s 453us/step - loss: 1.2099 -
accuracy: 0.5800
Epoch 23/100
2000/2000 [==============================] - 1s 525us/step - loss: 1.2285 -
accuracy: 0.5840
Epoch 24/100
2000/2000 [==============================] - 1s 481us/step - loss: 1.2098 -
accuracy: 0.5900
Epoch 25/100
2000/2000 [==============================] - 1s 520us/step - loss: 1.1601 -
accuracy: 0.5970
Epoch 26/100
2000/2000 [==============================] - 1s 437us/step - loss: 1.1339 -
accuracy: 0.61351s - los
Epoch 27/100
2000/2000 [==============================] - 1s 406us/step - loss: 1.1739 -
accuracy: 0.5995
Epoch 28/100
2000/2000 [==============================] - 1s 457us/step - loss: 1.1355 -
accuracy: 0.6145
Epoch 29/100
2000/2000 [==============================] - 1s 429us/step - loss: 1.1257 -
accuracy: 0.6210
Epoch 30/100
2000/2000 [==============================] - 1s 392us/step - loss: 1.0542 -
accuracy: 0.6335
Epoch 31/100
2000/2000 [==============================] - 1s 379us/step - loss: 1.0543 -
accuracy: 0.6280
Epoch 32/100
2000/2000 [==============================] - 1s 451us/step - loss: 1.0403 -
accuracy: 0.6440
Epoch 33/100
2000/2000 [==============================] - 1s 444us/step - loss: 1.0090 -
accuracy: 0.6635
Epoch 34/100
2000/2000 [==============================] - 1s 384us/step - loss: 1.0615 -
accuracy: 0.6285
Epoch 35/100
2000/2000 [==============================] - 1s 380us/step - loss: 0.9469 -
```

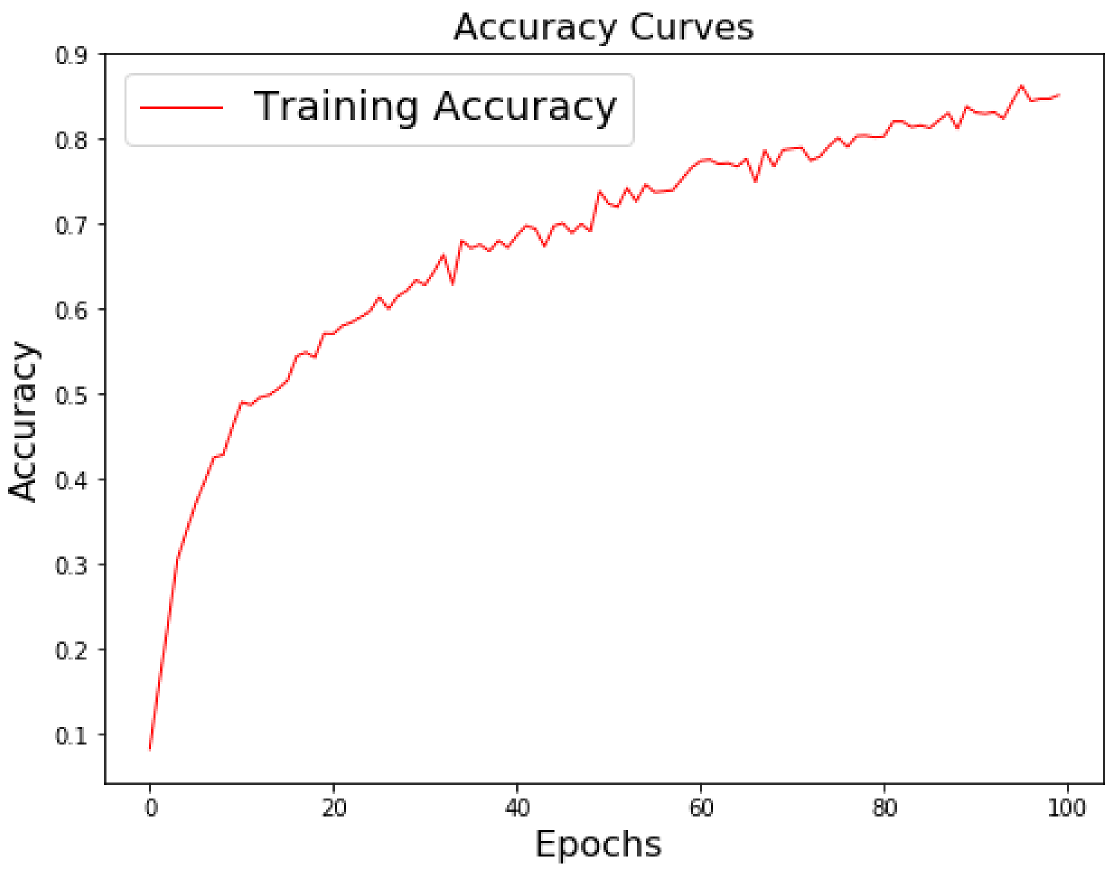
```
accuracy: 0.6800
Epoch 36/100
2000/2000 [==============================] - 1s 369us/step - loss: 0.9668 -
accuracy: 0.6710
Epoch 37/100
2000/2000 [==============================] - 1s 394us/step - loss: 0.9375 -
accuracy: 0.6750
Epoch 38/100
2000/2000 [==============================] - 1s 396us/step - loss: 0.9546 -
accuracy: 0.6675
Epoch 39/100
2000/2000 [==============================] - 1s 379us/step - loss: 0.9684 -
accuracy: 0.6800
Epoch 40/100
2000/2000 [==============================] - 1s 368us/step - loss: 0.9604 -
accuracy: 0.6715
Epoch 41/100
2000/2000 [==============================] - 1s 357us/step - loss: 0.9103 -
accuracy: 0.6860
Epoch 42/100
2000/2000 [==============================] - 1s 368us/step - loss: 0.9036 -
accuracy: 0.6975
Epoch 43/100
2000/2000 [==============================] - 1s 361us/step - loss: 0.9089 -
accuracy: 0.6940
Epoch 44/100
2000/2000 [==============================] - 1s 369us/step - loss: 0.8919 -
accuracy: 0.6730
Epoch 45/100
2000/2000 [==============================] - 1s 371us/step - loss: 0.8685 -
accuracy: 0.6970
Epoch 46/100
2000/2000 [==============================] - 1s 407us/step - loss: 0.8901 -
accuracy: 0.7005
Epoch 47/100
2000/2000 [==============================] - 1s 401us/step - loss: 0.8926 -
accuracy: 0.6890
Epoch 48/100
2000/2000 [==============================] - 1s 500us/step - loss: 0.8862 -
accuracy: 0.6995
Epoch 49/100
2000/2000 [==============================] - 1s 367us/step - loss: 0.8689 -
accuracy: 0.6910
Epoch 50/100
2000/2000 [==============================] - 1s 378us/step - loss: 0.7855 -
accuracy: 0.7380
Epoch 51/100
2000/2000 [==============================] - 1s 361us/step - loss: 0.8092 -
```

```
accuracy: 0.7230
Epoch 52/100
2000/2000 [==============================] - 1s 381us/step - loss: 0.7991 -
accuracy: 0.7195
Epoch 53/100
2000/2000 [==============================] - 1s 382us/step - loss: 0.7619 -
accuracy: 0.7420
Epoch 54/100
2000/2000 [==============================] - 1s 375us/step - loss: 0.7942 -
accuracy: 0.7260
Epoch 55/100
2000/2000 [==============================] - 1s 382us/step - loss: 0.7489 -
accuracy: 0.7460
Epoch 56/100
2000/2000 [==============================] - 1s 373us/step - loss: 0.7600 -
accuracy: 0.7370
Epoch 57/100
2000/2000 [==============================] - 1s 384us/step - loss: 0.7641 -
accuracy: 0.7380
Epoch 58/100
2000/2000 [==============================] - 1s 382us/step - loss: 0.7701 -
accuracy: 0.7395
Epoch 59/100
2000/2000 [==============================] - 1s 376us/step - loss: 0.7360 -
accuracy: 0.7525
Epoch 60/100
2000/2000 [==============================] - 1s 348us/step - loss: 0.6973 -
accuracy: 0.7655
Epoch 61/100
2000/2000 [==============================] - 1s 367us/step - loss: 0.6461 -
accuracy: 0.7735
Epoch 62/100
2000/2000 [==============================] - 1s 359us/step - loss: 0.6645 -
accuracy: 0.7750
Epoch 63/100
2000/2000 [==============================] - 1s 368us/step - loss: 0.6883 -
accuracy: 0.7700
Epoch 64/100
2000/2000 [==============================] - 1s 361us/step - loss: 0.6795 -
accuracy: 0.7710
Epoch 65/100
2000/2000 [==============================] - 1s 368us/step - loss: 0.6812 -
accuracy: 0.7670
Epoch 66/100
2000/2000 [==============================] - 1s 372us/step - loss: 0.6516 -
accuracy: 0.7760
Epoch 67/100
2000/2000 [==============================] - 1s 354us/step - loss: 0.7090 -
```

```
accuracy: 0.7490
Epoch 68/100
2000/2000 [==============================] - 1s 360us/step - loss: 0.6413 -
accuracy: 0.7860
Epoch 69/100
2000/2000 [==============================] - 1s 356us/step - loss: 0.6740 -
accuracy: 0.7670
Epoch 70/100
2000/2000 [==============================] - 1s 364us/step - loss: 0.6338 -
accuracy: 0.7865
Epoch 71/100
2000/2000 [==============================] - 1s 358us/step - loss: 0.6280 -
accuracy: 0.7880
Epoch 72/100
2000/2000 [==============================] - 1s 369us/step - loss: 0.6056 -
accuracy: 0.7895
Epoch 73/100
2000/2000 [==============================] - 1s 351us/step - loss: 0.6123 -
accuracy: 0.7740
Epoch 74/100
2000/2000 [==============================] - 1s 370us/step - loss: 0.6309 -
accuracy: 0.7785
Epoch 75/100
2000/2000 [==============================] - 1s 361us/step - loss: 0.5931 -
accuracy: 0.7910
Epoch 76/100
2000/2000 [==============================] - 1s 365us/step - loss: 0.5721 -
accuracy: 0.8010
Epoch 77/100
2000/2000 [==============================] - 1s 357us/step - loss: 0.6132 -
accuracy: 0.7900
Epoch 78/100
2000/2000 [==============================] - 1s 362us/step - loss: 0.5771 -
accuracy: 0.8030
Epoch 79/100
2000/2000 [==============================] - 1s 352us/step - loss: 0.5568 -
accuracy: 0.8035
Epoch 80/100
2000/2000 [==============================] - 1s 355us/step - loss: 0.5645 -
accuracy: 0.8015
Epoch 81/100
2000/2000 [==============================] - 1s 459us/step - loss: 0.5923 -
accuracy: 0.8025
Epoch 82/100
2000/2000 [==============================] - 1s 518us/step - loss: 0.5361 -
accuracy: 0.8200
Epoch 83/100
2000/2000 [==============================] - 1s 451us/step - loss: 0.5231 -
```

```
accuracy: 0.8200
Epoch 84/100
2000/2000 [==============================] - 1s 425us/step - loss: 0.5318 -
accuracy: 0.8135
Epoch 85/100
2000/2000 [==============================] - 1s 412us/step - loss: 0.5263 -
accuracy: 0.8155
Epoch 86/100
2000/2000 [==============================] - 1s 422us/step - loss: 0.5555 -
accuracy: 0.8125
Epoch 87/100
2000/2000 [==============================] - 1s 438us/step - loss: 0.5357 -
accuracy: 0.8215
Epoch 88/100
2000/2000 [==============================] - 1s 475us/step - loss: 0.5030 -
accuracy: 0.8300
Epoch 89/100
2000/2000 [==============================] - 1s 462us/step - loss: 0.5311 -
accuracy: 0.8115
Epoch 90/100
2000/2000 [==============================] - 1s 431us/step - loss: 0.4658 -
accuracy: 0.8375
Epoch 91/100
2000/2000 [==============================] - 1s 387us/step - loss: 0.5081 -
accuracy: 0.8305
Epoch 92/100
2000/2000 [==============================] - 1s 387us/step - loss: 0.5035 -
accuracy: 0.8290
Epoch 93/100
2000/2000 [==============================] - 1s 436us/step - loss: 0.4983 -
accuracy: 0.8310
Epoch 94/100
2000/2000 [==============================] - 1s 371us/step - loss: 0.5443 -
accuracy: 0.8235
Epoch 95/100
2000/2000 [==============================] - 1s 442us/step - loss: 0.4805 -
accuracy: 0.8440
Epoch 96/100
2000/2000 [==============================] - 1s 387us/step - loss: 0.4330 -
accuracy: 0.8625
Epoch 97/100
2000/2000 [==============================] - 1s 380us/step - loss: 0.4566 -
accuracy: 0.8440
Epoch 98/100
2000/2000 [==============================] - 1s 346us/step - loss: 0.4458 -
accuracy: 0.8465
Epoch 99/100
2000/2000 [==============================] - 1s 413us/step - loss: 0.4432 -
```

```
accuracy: 0.8465
Epoch 100/100
2000/2000 [==============================] - 1s 366us/step - loss: 0.4234 -
accuracy: 0.8510
training  model takes 91.207 seconds
```



[135]:
```python
t0 = time.time()
pred2 = model2.predict(test_data)
pred_lst2 = []
for i in range(len(pred2)):
    arr = pred2[i]
    idx = np.argwhere(arr == np.max(arr))
    pred_lst2.append(idx[0][0])
tst_labl = np.argmax(test_label_cat, axis=-1)
acc = accuracy_score(pred_lst2, tst_labl)
print("Test accuracy is %s percent" %(acc*100))
print("testing model takes %s seconds" % round((time.time() - t0),3))
```

```
Test accuracy is 54.400000000000006 percent
testing model takes 0.125 seconds
```

2.1 Read the testing data

```
[ ]: # read mat file and store coordinates in mat

     test_dir = os.path.join(root,  '../data/XXXX')
     test_path =  os.path.join(test_dir,  "label.csv")
     test_index = pd.read_csv(test_path)
     test_pt_dir =  os.path.join(test_dir, 'points' )

     m_test = []
     for idx in test_index['Index']:
         file = "%04d.mat"%(idx)
         m_test.append( scipy.io.loadmat( os.path.join( test_pt_dir, file ) ))
```

```
[ ]: pairwise_dist_cal_updt(m_test)
```