Final Model - Group4

October 31, 2019

0.0.1 Importing required packages

[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
```

```python
from keras.optimizers import Adam
```

0.0.2 loading data

[3]:
```python
# 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]:
```python
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]
```

0.0.3 train test split

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

[6]:
```python
train_mat = [ mat[i-1] for i in train_idx ]
test_mat = [ mat[i-1] for i in test_idx ]
```

[7]:
```python
train_labels = info.emotion_idx[train_idx-1]
test_labels = info.emotion_idx[test_idx-1]
```

[8]:
```python
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:]
```

0.0.4 feature extraction

[9]:
```
# 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])
```

[10]:
```
print("training: ")
train_data = pairwise_dist_cal_updt(train_mat[0:])
```

```
training:
feature constructions takes 19.490326166152954 seconds
```

[11]:
```
print("testing: ")
test_data = pairwise_dist_cal_updt(test_mat[0:])
```

```
testing:
feature constructions takes 4.743052959442139 seconds
```

0.0.5 XGB

[12]:
```
from xgboost.sklearn import XGBClassifier
```

[13]:
```
train_labels_xgb = [ x  - 1 for x in train_labels ]
test_labels_xgb = [ x  - 1 for x in test_labels ]
```

[30]:
```
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)
start_time=time.time()
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 1156.417 seconds
```

[31]:
```
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 1.199 seconds
```

[32]:
```
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

[33]:
```
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)
```

[34]:
```
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))
```

```
training  model takes 2236.248 seconds
```

[40]:
```
pred_xgb_2_train = xgb_2.predict(train_data)
acc_2_train = accuracy_score(pred_xgb_2_train,train_labels_xgb )
print("Train accuracy is %s percent" %(acc_2_train *100))
```

```
Train accuracy is 98.15 percent
```

[41]:
```
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 2.033 seconds
```

[42]:
```
acc_2_test = accuracy_score(pred_xgb_2,test_labels_xgb )
print("Test accuracy is %s percent" %(acc_2_test *100))
```

```
Test accuracy is 49.2 percent
```

Since XGB doesn't improve that much. We deceided to try other models, like Neural Network

0.0.6 Neural Network

[14]:
```
X = train_data
Y = train_label_cat
```

first model

[15]:
```
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)
```

[16]:
```
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))
```

```
WARNING:tensorflow:From /Users/Qiqi/opt/anaconda3/lib/python3.7/site-
packages/keras/backend/tensorflow_backend.py:422: The name tf.global_variables
is deprecated. Please use tf.compat.v1.global_variables instead.

Epoch 1/100
2000/2000 [==============================] - 3s 1ms/step - loss: 3.0098 -
accuracy: 0.1100
Epoch 2/100
2000/2000 [==============================] - 2s 1ms/step - loss: 2.6026 -
accuracy: 0.1940
Epoch 3/100
2000/2000 [==============================] - 2s 1ms/step - loss: 2.3021 -
accuracy: 0.2700
Epoch 4/100
2000/2000 [==============================] - 2s 1ms/step - loss: 2.1436 -
accuracy: 0.2975
Epoch 5/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.9974 -
accuracy: 0.3440
Epoch 6/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.8468 -
accuracy: 0.3920
Epoch 7/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.8075 -
accuracy: 0.4035
Epoch 8/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.6632 -
accuracy: 0.4440
Epoch 9/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.6180 -
accuracy: 0.4590
Epoch 10/100
2000/2000 [==============================] - 2s 981us/step - loss: 1.5304 -
accuracy: 0.4800
Epoch 11/100
2000/2000 [==============================] - 2s 987us/step - loss: 1.4909 -
accuracy: 0.5015
Epoch 12/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.4268 -
accuracy: 0.5180
Epoch 13/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.3894 -
accuracy: 0.5365
Epoch 14/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.3802 -
accuracy: 0.5255
Epoch 15/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.3244 -
```

```
accuracy: 0.5605
Epoch 16/100
2000/2000 [==============================] - 3s 1ms/step - loss: 1.2694 -
accuracy: 0.5745
Epoch 17/100
2000/2000 [==============================] - 2s 933us/step - loss: 1.2331 -
accuracy: 0.5825
Epoch 18/100
2000/2000 [==============================] - 2s 1000us/step - loss: 1.2198 -
accuracy: 0.5815
Epoch 19/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.1474 -
accuracy: 0.5970
Epoch 20/100
2000/2000 [==============================] - 3s 1ms/step - loss: 1.1679 -
accuracy: 0.6030
Epoch 21/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.1470 -
accuracy: 0.6075
Epoch 22/100
2000/2000 [==============================] - 2s 965us/step - loss: 1.0993 -
accuracy: 0.6320
Epoch 23/100
2000/2000 [==============================] - 2s 967us/step - loss: 1.1024 -
accuracy: 0.6235
Epoch 24/100
2000/2000 [==============================] - 2s 956us/step - loss: 1.0572 -
accuracy: 0.6230
Epoch 25/100
2000/2000 [==============================] - 2s 907us/step - loss: 1.0434 -
accuracy: 0.6440
Epoch 26/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.0149 -
accuracy: 0.6510
Epoch 27/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.9574 -
accuracy: 0.6670
Epoch 28/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.9542 -
accuracy: 0.6605
Epoch 29/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.9491 -
accuracy: 0.6785
Epoch 30/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.9717 -
accuracy: 0.6615
Epoch 31/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.9332 -
```

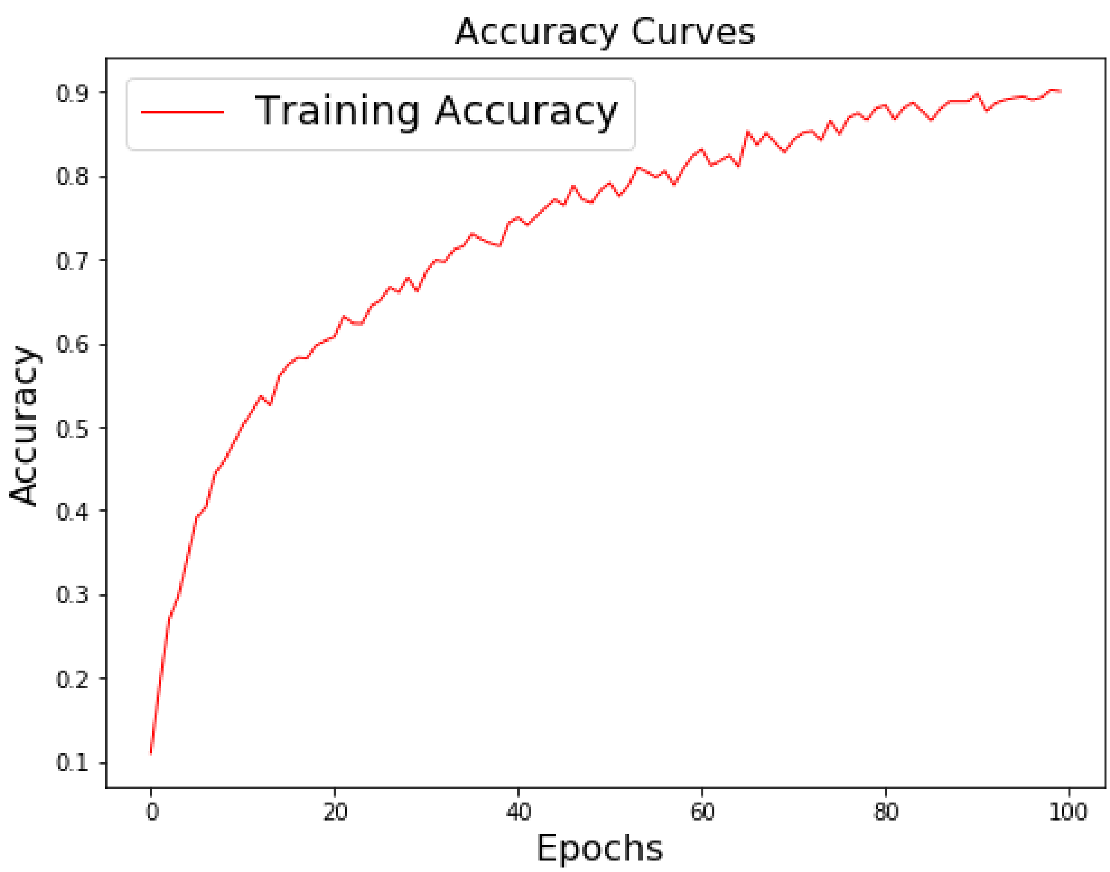
```
accuracy: 0.6855
Epoch 32/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.8520 -
accuracy: 0.6990
Epoch 33/100
2000/2000 [==============================] - 2s 980us/step - loss: 0.9046 -
accuracy: 0.6970
Epoch 34/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.8415 -
accuracy: 0.7115
Epoch 35/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.8357 -
accuracy: 0.7155
Epoch 36/100
2000/2000 [==============================] - 2s 947us/step - loss: 0.8008 -
accuracy: 0.7305
Epoch 37/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.7911 -
accuracy: 0.7240
Epoch 38/100
2000/2000 [==============================] - 2s 979us/step - loss: 0.8218 -
accuracy: 0.7190
Epoch 39/100
2000/2000 [==============================] - 2s 941us/step - loss: 0.8060 -
accuracy: 0.7160
Epoch 40/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.7463 -
accuracy: 0.7435
Epoch 41/100
2000/2000 [==============================] - 2s 912us/step - loss: 0.7460 -
accuracy: 0.7500
Epoch 42/100
2000/2000 [==============================] - 2s 869us/step - loss: 0.7506 -
accuracy: 0.7410
Epoch 43/100
2000/2000 [==============================] - 2s 878us/step - loss: 0.7058 -
accuracy: 0.7515
Epoch 44/100
2000/2000 [==============================] - 2s 886us/step - loss: 0.6827 -
accuracy: 0.7620
Epoch 45/100
2000/2000 [==============================] - 2s 885us/step - loss: 0.6540 -
accuracy: 0.7715
Epoch 46/100
2000/2000 [==============================] - 2s 886us/step - loss: 0.6767 -
accuracy: 0.7645
Epoch 47/100
2000/2000 [==============================] - 2s 883us/step - loss: 0.6161 -
```

```
accuracy: 0.7880
Epoch 48/100
2000/2000 [==============================] - 2s 898us/step - loss: 0.6803 -
accuracy: 0.7715
Epoch 49/100
2000/2000 [==============================] - 2s 938us/step - loss: 0.6737 -
accuracy: 0.7675
Epoch 50/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.6390 -
accuracy: 0.7830
Epoch 51/100
2000/2000 [==============================] - 2s 894us/step - loss: 0.6277 -
accuracy: 0.7915
Epoch 52/100
2000/2000 [==============================] - 2s 854us/step - loss: 0.6583 -
accuracy: 0.7755
Epoch 53/100
2000/2000 [==============================] - 2s 855us/step - loss: 0.6178 -
accuracy: 0.7880
Epoch 54/100
2000/2000 [==============================] - 2s 844us/step - loss: 0.5641 -
accuracy: 0.8095
Epoch 55/100
2000/2000 [==============================] - 2s 839us/step - loss: 0.5681 -
accuracy: 0.8045
Epoch 56/100
2000/2000 [==============================] - 2s 852us/step - loss: 0.5861 -
accuracy: 0.7980
Epoch 57/100
2000/2000 [==============================] - 2s 851us/step - loss: 0.5702 -
accuracy: 0.8055
Epoch 58/100
2000/2000 [==============================] - 2s 856us/step - loss: 0.6220 -
accuracy: 0.7885
Epoch 59/100
2000/2000 [==============================] - 2s 864us/step - loss: 0.5483 -
accuracy: 0.8085
Epoch 60/100
2000/2000 [==============================] - 2s 855us/step - loss: 0.5028 -
accuracy: 0.8235
Epoch 61/100
2000/2000 [==============================] - 2s 849us/step - loss: 0.4832 -
accuracy: 0.8315
Epoch 62/100
2000/2000 [==============================] - 2s 846us/step - loss: 0.5437 -
accuracy: 0.8120
Epoch 63/100
2000/2000 [==============================] - 2s 848us/step - loss: 0.5407 -
```

```
accuracy: 0.8175
Epoch 64/100
2000/2000 [==============================] - 2s 860us/step - loss: 0.5417 -
accuracy: 0.8240
Epoch 65/100
2000/2000 [==============================] - 2s 863us/step - loss: 0.5588 -
accuracy: 0.8105
Epoch 66/100
2000/2000 [==============================] - 2s 861us/step - loss: 0.4595 -
accuracy: 0.8530
Epoch 67/100
2000/2000 [==============================] - 2s 852us/step - loss: 0.4676 -
accuracy: 0.8365
Epoch 68/100
2000/2000 [==============================] - 2s 848us/step - loss: 0.4443 -
accuracy: 0.8510
Epoch 69/100
2000/2000 [==============================] - 2s 855us/step - loss: 0.4556 -
accuracy: 0.8395
Epoch 70/100
2000/2000 [==============================] - 2s 902us/step - loss: 0.4938 -
accuracy: 0.8275
Epoch 71/100
2000/2000 [==============================] - 2s 868us/step - loss: 0.4749 -
accuracy: 0.8430
Epoch 72/100
2000/2000 [==============================] - 2s 867us/step - loss: 0.4462 -
accuracy: 0.8510
Epoch 73/100
2000/2000 [==============================] - 2s 870us/step - loss: 0.4206 -
accuracy: 0.8530
Epoch 74/100
2000/2000 [==============================] - 2s 884us/step - loss: 0.4645 -
accuracy: 0.8425
Epoch 75/100
2000/2000 [==============================] - 2s 874us/step - loss: 0.4099 -
accuracy: 0.8655
Epoch 76/100
2000/2000 [==============================] - 2s 840us/step - loss: 0.4463 -
accuracy: 0.8495
Epoch 77/100
2000/2000 [==============================] - 2s 832us/step - loss: 0.4053 -
accuracy: 0.8695
Epoch 78/100
2000/2000 [==============================] - 2s 850us/step - loss: 0.3804 -
accuracy: 0.8745
Epoch 79/100
2000/2000 [==============================] - 2s 845us/step - loss: 0.4179 -
```

```
accuracy: 0.8665
Epoch 80/100
2000/2000 [==============================] - 2s 841us/step - loss: 0.3642 -
accuracy: 0.8805
Epoch 81/100
2000/2000 [==============================] - 2s 836us/step - loss: 0.3640 -
accuracy: 0.8840
Epoch 82/100
2000/2000 [==============================] - 2s 846us/step - loss: 0.3795 -
accuracy: 0.8670
Epoch 83/100
2000/2000 [==============================] - 2s 852us/step - loss: 0.3688 -
accuracy: 0.8810
Epoch 84/100
2000/2000 [==============================] - 2s 869us/step - loss: 0.3296 -
accuracy: 0.8870
Epoch 85/100
2000/2000 [==============================] - 2s 853us/step - loss: 0.3453 -
accuracy: 0.8770
Epoch 86/100
2000/2000 [==============================] - 2s 850us/step - loss: 0.4187 -
accuracy: 0.8660
Epoch 87/100
2000/2000 [==============================] - 2s 856us/step - loss: 0.3666 -
accuracy: 0.8800
Epoch 88/100
2000/2000 [==============================] - 2s 866us/step - loss: 0.3456 -
accuracy: 0.8885
Epoch 89/100
2000/2000 [==============================] - 2s 868us/step - loss: 0.3607 -
accuracy: 0.8885
Epoch 90/100
2000/2000 [==============================] - 2s 867us/step - loss: 0.3411 -
accuracy: 0.8885
Epoch 91/100
2000/2000 [==============================] - 2s 868us/step - loss: 0.3199 -
accuracy: 0.8980
Epoch 92/100
2000/2000 [==============================] - 2s 874us/step - loss: 0.3638 -
accuracy: 0.8770
Epoch 93/100
2000/2000 [==============================] - 2s 867us/step - loss: 0.3461 -
accuracy: 0.8865
Epoch 94/100
2000/2000 [==============================] - 2s 870us/step - loss: 0.3559 -
accuracy: 0.8905
Epoch 95/100
2000/2000 [==============================] - 2s 875us/step - loss: 0.3377 -
```

```
accuracy: 0.8930
Epoch 96/100
2000/2000 [==============================] - 2s 880us/step - loss: 0.3310 -
accuracy: 0.8945
Epoch 97/100
2000/2000 [==============================] - 2s 947us/step - loss: 0.3237 -
accuracy: 0.8905
Epoch 98/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3168 -
accuracy: 0.8935
Epoch 99/100
2000/2000 [==============================] - 2s 904us/step - loss: 0.3273 -
accuracy: 0.9025
Epoch 100/100
2000/2000 [==============================] - 2s 866us/step - loss: 0.2955 -
accuracy: 0.9010
training  model takes 195.464 seconds
```



[45]:
```
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" % round(acc*100,3))
print("testing model takes %s seconds" % round((time.time() - t0),3))
```

```
Test accuracy is 53.6 percent
testing model takes 0.144 seconds
```

second model

[18]:
```
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)
```

[19]:
```
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 [==============================] - 1s 690us/step - loss: 3.0590 -
accuracy: 0.0815
Epoch 2/100
2000/2000 [==============================] - 1s 399us/step - loss: 2.7153 -
accuracy: 0.1690
Epoch 3/100
2000/2000 [==============================] - 1s 413us/step - loss: 2.3441 -
accuracy: 0.2405
Epoch 4/100
2000/2000 [==============================] - 1s 409us/step - loss: 2.1477 -
accuracy: 0.2965
Epoch 5/100
2000/2000 [==============================] - 1s 445us/step - loss: 1.9895 -
accuracy: 0.3360
Epoch 6/100
2000/2000 [==============================] - 1s 415us/step - loss: 1.8939 -
accuracy: 0.3710
Epoch 7/100
2000/2000 [==============================] - 1s 399us/step - loss: 1.8041 -
accuracy: 0.3915
Epoch 8/100
2000/2000 [==============================] - 1s 399us/step - loss: 1.7150 -
accuracy: 0.4380
Epoch 9/100
2000/2000 [==============================] - 1s 395us/step - loss: 1.6383 -
accuracy: 0.4565
Epoch 10/100
2000/2000 [==============================] - 1s 433us/step - loss: 1.6166 -
accuracy: 0.4625
Epoch 11/100
2000/2000 [==============================] - 1s 406us/step - loss: 1.5693 -
accuracy: 0.4705
Epoch 12/100
2000/2000 [==============================] - 1s 408us/step - loss: 1.5100 -
accuracy: 0.4810
Epoch 13/100
2000/2000 [==============================] - 1s 404us/step - loss: 1.4937 -
accuracy: 0.4740
Epoch 14/100
2000/2000 [==============================] - 1s 416us/step - loss: 1.4503 -
accuracy: 0.5055
Epoch 15/100
2000/2000 [==============================] - 1s 415us/step - loss: 1.3912 -
accuracy: 0.5140
Epoch 16/100
2000/2000 [==============================] - 1s 407us/step - loss: 1.3669 -
accuracy: 0.5355
Epoch 17/100
```

```
2000/2000 [==============================] - 1s 424us/step - loss: 1.3558 -
accuracy: 0.5405
Epoch 18/100
2000/2000 [==============================] - 1s 396us/step - loss: 1.3229 -
accuracy: 0.5545
Epoch 19/100
2000/2000 [==============================] - 1s 408us/step - loss: 1.3107 -
accuracy: 0.5550
Epoch 20/100
2000/2000 [==============================] - 1s 432us/step - loss: 1.3305 -
accuracy: 0.5415
Epoch 21/100
2000/2000 [==============================] - 1s 462us/step - loss: 1.2595 -
accuracy: 0.5740
Epoch 22/100
2000/2000 [==============================] - 1s 408us/step - loss: 1.2346 -
accuracy: 0.5710
Epoch 23/100
2000/2000 [==============================] - 1s 409us/step - loss: 1.1947 -
accuracy: 0.5960
Epoch 24/100
2000/2000 [==============================] - 1s 423us/step - loss: 1.1987 -
accuracy: 0.6010
Epoch 25/100
2000/2000 [==============================] - 1s 438us/step - loss: 1.1820 -
accuracy: 0.5935
Epoch 26/100
2000/2000 [==============================] - 1s 432us/step - loss: 1.1559 -
accuracy: 0.5955
Epoch 27/100
2000/2000 [==============================] - 1s 418us/step - loss: 1.1476 -
accuracy: 0.6075
Epoch 28/100
2000/2000 [==============================] - 1s 420us/step - loss: 1.1151 -
accuracy: 0.6210
Epoch 29/100
2000/2000 [==============================] - 1s 408us/step - loss: 1.1068 -
accuracy: 0.6195
Epoch 30/100
2000/2000 [==============================] - 1s 420us/step - loss: 1.0692 -
accuracy: 0.6355
Epoch 31/100
2000/2000 [==============================] - 1s 421us/step - loss: 1.0720 -
accuracy: 0.6355
Epoch 32/100
2000/2000 [==============================] - 1s 413us/step - loss: 1.0374 -
accuracy: 0.6410
Epoch 33/100
```

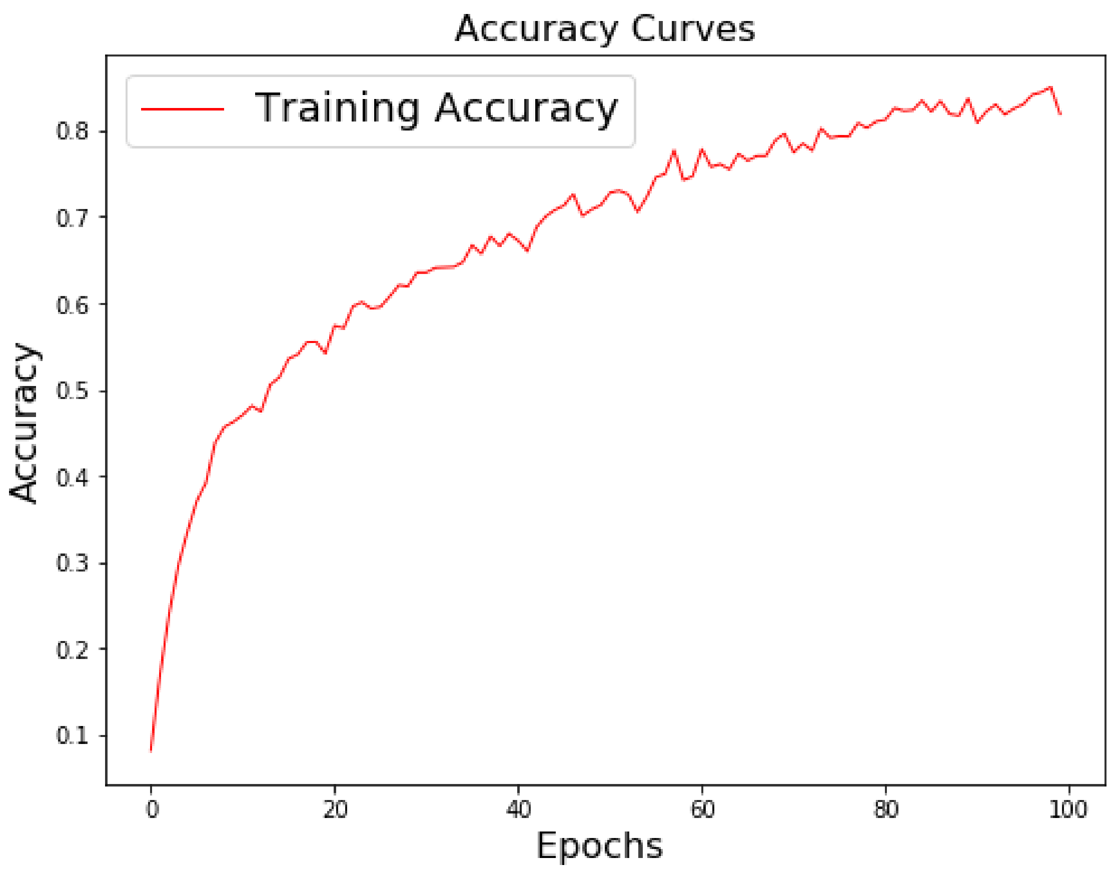
```
2000/2000 [==============================] - 1s 438us/step - loss: 1.0714 -
accuracy: 0.6415
Epoch 34/100
2000/2000 [==============================] - 1s 443us/step - loss: 1.0232 -
accuracy: 0.6420
Epoch 35/100
2000/2000 [==============================] - 1s 440us/step - loss: 1.0210 -
accuracy: 0.6475
Epoch 36/100
2000/2000 [==============================] - 1s 422us/step - loss: 0.9603 -
accuracy: 0.6670
Epoch 37/100
2000/2000 [==============================] - 1s 408us/step - loss: 0.9819 -
accuracy: 0.6570
Epoch 38/100
2000/2000 [==============================] - 1s 405us/step - loss: 0.9750 -
accuracy: 0.6775
Epoch 39/100
2000/2000 [==============================] - 1s 429us/step - loss: 0.9673 -
accuracy: 0.6660
Epoch 40/100
2000/2000 [==============================] - 1s 422us/step - loss: 0.9188 -
accuracy: 0.6805
Epoch 41/100
2000/2000 [==============================] - 1s 409us/step - loss: 0.9627 -
accuracy: 0.6725
Epoch 42/100
2000/2000 [==============================] - 1s 414us/step - loss: 0.9577 -
accuracy: 0.6600
Epoch 43/100
2000/2000 [==============================] - 1s 422us/step - loss: 0.9035 -
accuracy: 0.6880
Epoch 44/100
2000/2000 [==============================] - 1s 416us/step - loss: 0.8769 -
accuracy: 0.7005
Epoch 45/100
2000/2000 [==============================] - 1s 418us/step - loss: 0.8466 -
accuracy: 0.7080
Epoch 46/100
2000/2000 [==============================] - 1s 447us/step - loss: 0.8414 -
accuracy: 0.7130
Epoch 47/100
2000/2000 [==============================] - 1s 413us/step - loss: 0.7949 -
accuracy: 0.7265
Epoch 48/100
2000/2000 [==============================] - 1s 420us/step - loss: 0.8681 -
accuracy: 0.7010
Epoch 49/100
```

```
2000/2000 [==============================] - 1s 430us/step - loss: 0.8371 -
accuracy: 0.7085
Epoch 50/100
2000/2000 [==============================] - 1s 424us/step - loss: 0.8439 -
accuracy: 0.7135
Epoch 51/100
2000/2000 [==============================] - 1s 429us/step - loss: 0.7993 -
accuracy: 0.7280
Epoch 52/100
2000/2000 [==============================] - 1s 424us/step - loss: 0.7692 -
accuracy: 0.7305
Epoch 53/100
2000/2000 [==============================] - 1s 467us/step - loss: 0.7951 -
accuracy: 0.7260
Epoch 54/100
2000/2000 [==============================] - 1s 441us/step - loss: 0.8462 -
accuracy: 0.7055
Epoch 55/100
2000/2000 [==============================] - 1s 435us/step - loss: 0.8228 -
accuracy: 0.7230
Epoch 56/100
2000/2000 [==============================] - 1s 424us/step - loss: 0.7397 -
accuracy: 0.7460
Epoch 57/100
2000/2000 [==============================] - 1s 440us/step - loss: 0.7218 -
accuracy: 0.7495
Epoch 58/100
2000/2000 [==============================] - 1s 437us/step - loss: 0.6724 -
accuracy: 0.7770
Epoch 59/100
2000/2000 [==============================] - 1s 426us/step - loss: 0.7428 -
accuracy: 0.7425
Epoch 60/100
2000/2000 [==============================] - 1s 433us/step - loss: 0.7412 -
accuracy: 0.7470
Epoch 61/100
2000/2000 [==============================] - 1s 433us/step - loss: 0.6528 -
accuracy: 0.7785
Epoch 62/100
2000/2000 [==============================] - 1s 432us/step - loss: 0.7069 -
accuracy: 0.7575
Epoch 63/100
2000/2000 [==============================] - 1s 438us/step - loss: 0.6897 -
accuracy: 0.7610
Epoch 64/100
2000/2000 [==============================] - 1s 426us/step - loss: 0.7245 -
accuracy: 0.7550
Epoch 65/100
```

```
2000/2000 [==============================] - 1s 449us/step - loss: 0.6653 -
accuracy: 0.7730
Epoch 66/100
2000/2000 [==============================] - 1s 438us/step - loss: 0.6764 -
accuracy: 0.7650
Epoch 67/100
2000/2000 [==============================] - 1s 430us/step - loss: 0.6674 -
accuracy: 0.7705
Epoch 68/100
2000/2000 [==============================] - 1s 435us/step - loss: 0.6639 -
accuracy: 0.7705
Epoch 69/100
2000/2000 [==============================] - 1s 430us/step - loss: 0.6315 -
accuracy: 0.7885
Epoch 70/100
2000/2000 [==============================] - 1s 432us/step - loss: 0.6171 -
accuracy: 0.7965
Epoch 71/100
2000/2000 [==============================] - 1s 429us/step - loss: 0.6642 -
accuracy: 0.7745
Epoch 72/100
2000/2000 [==============================] - 1s 423us/step - loss: 0.6131 -
accuracy: 0.7850
Epoch 73/100
2000/2000 [==============================] - 1s 438us/step - loss: 0.6445 -
accuracy: 0.7765
Epoch 74/100
2000/2000 [==============================] - 1s 426us/step - loss: 0.5957 -
accuracy: 0.8025
Epoch 75/100
2000/2000 [==============================] - 1s 441us/step - loss: 0.6373 -
accuracy: 0.7915
Epoch 76/100
2000/2000 [==============================] - 1s 439us/step - loss: 0.6218 -
accuracy: 0.7935
Epoch 77/100
2000/2000 [==============================] - 1s 450us/step - loss: 0.6312 -
accuracy: 0.7930
Epoch 78/100
2000/2000 [==============================] - 1s 406us/step - loss: 0.5746 -
accuracy: 0.8085
Epoch 79/100
2000/2000 [==============================] - 1s 427us/step - loss: 0.5848 -
accuracy: 0.8030
Epoch 80/100
2000/2000 [==============================] - 1s 741us/step - loss: 0.5477 -
accuracy: 0.8105
Epoch 81/100
```

```
2000/2000 [==============================] - 1s 741us/step - loss: 0.5349 -
accuracy: 0.8125
Epoch 82/100
2000/2000 [==============================] - 1s 651us/step - loss: 0.5381 -
accuracy: 0.8260
Epoch 83/100
2000/2000 [==============================] - 1s 605us/step - loss: 0.5320 -
accuracy: 0.8225
Epoch 84/100
2000/2000 [==============================] - 1s 707us/step - loss: 0.5029 -
accuracy: 0.8235
Epoch 85/100
2000/2000 [==============================] - 1s 669us/step - loss: 0.5050 -
accuracy: 0.8345
Epoch 86/100
2000/2000 [==============================] - 1s 599us/step - loss: 0.5244 -
accuracy: 0.8215
Epoch 87/100
2000/2000 [==============================] - 1s 549us/step - loss: 0.4986 -
accuracy: 0.8340
Epoch 88/100
2000/2000 [==============================] - 1s 527us/step - loss: 0.5296 -
accuracy: 0.8190
Epoch 89/100
2000/2000 [==============================] - 1s 601us/step - loss: 0.5135 -
accuracy: 0.8170
Epoch 90/100
2000/2000 [==============================] - 1s 638us/step - loss: 0.4818 -
accuracy: 0.8370
Epoch 91/100
2000/2000 [==============================] - 1s 510us/step - loss: 0.5363 -
accuracy: 0.8090
Epoch 92/100
2000/2000 [==============================] - 1s 639us/step - loss: 0.5282 -
accuracy: 0.8220
Epoch 93/100
2000/2000 [==============================] - 1s 656us/step - loss: 0.5103 -
accuracy: 0.8300
Epoch 94/100
2000/2000 [==============================] - 1s 645us/step - loss: 0.5158 -
accuracy: 0.8185
Epoch 95/100
2000/2000 [==============================] - 1s 507us/step - loss: 0.5069 -
accuracy: 0.8255
Epoch 96/100
2000/2000 [==============================] - 1s 604us/step - loss: 0.5065 -
accuracy: 0.8300
Epoch 97/100
```

```
2000/2000 [==============================] - 1s 558us/step - loss: 0.4514 -
accuracy: 0.8415
Epoch 98/100
2000/2000 [==============================] - 1s 510us/step - loss: 0.4713 -
accuracy: 0.8445
Epoch 99/100
2000/2000 [==============================] - 1s 533us/step - loss: 0.4496 -
accuracy: 0.8505
Epoch 100/100
2000/2000 [==============================] - 1s 368us/step - loss: 0.5042 -
accuracy: 0.8190
training  model takes 94.409 seconds
```



[43]:
```
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" % round(acc*100,3))
print("testing model takes %s seconds" % round((time.time() - t0),3))
```

```
Test accuracy is 56.4 percent
testing model takes 0.179 seconds
```

third model

[21]:
```
input_shape = [2450]
input_layer = Input(input_shape)
x = BatchNormalization(momentum = 0.88)(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)
model3 = Model(input_layer,output_layer)
```

[22]:
```
start_time = time.time()
model3.compile(loss='categorical_crossentropy',optimizer = Adam(lr=0.
 ↪001),metrics=['accuracy'])
model_his = model3.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 1ms/step - loss: 2.9908 -
accuracy: 0.0975
Epoch 2/100
2000/2000 [==============================] - 1s 750us/step - loss: 2.6035 -
accuracy: 0.2020
```

```
Epoch 3/100
2000/2000 [==============================] - 2s 1ms/step - loss: 2.2965 -
accuracy: 0.2825
Epoch 4/100
2000/2000 [==============================] - 2s 1ms/step - loss: 2.0985 -
accuracy: 0.3215
Epoch 5/100
2000/2000 [==============================] - 3s 1ms/step - loss: 1.9475 -
accuracy: 0.3640
Epoch 6/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.8338 -
accuracy: 0.3995
Epoch 7/100
2000/2000 [==============================] - 4s 2ms/step - loss: 1.7790 -
accuracy: 0.4050
Epoch 8/100
2000/2000 [==============================] - 3s 1ms/step - loss: 1.6514 -
accuracy: 0.4510
Epoch 9/100
2000/2000 [==============================] - 3s 1ms/step - loss: 1.5575 -
accuracy: 0.4695
Epoch 10/100
2000/2000 [==============================] - 3s 1ms/step - loss: 1.5226 -
accuracy: 0.4700
Epoch 11/100
2000/2000 [==============================] - 3s 2ms/step - loss: 1.4856 -
accuracy: 0.4965
Epoch 12/100
2000/2000 [==============================] - 3s 1ms/step - loss: 1.4759 -
accuracy: 0.4970
Epoch 13/100
2000/2000 [==============================] - 3s 1ms/step - loss: 1.4046 -
accuracy: 0.5185
Epoch 14/100
2000/2000 [==============================] - 3s 2ms/step - loss: 1.2936 -
accuracy: 0.5600
Epoch 15/100
2000/2000 [==============================] - 3s 2ms/step - loss: 1.3201 -
accuracy: 0.5510
Epoch 16/100
2000/2000 [==============================] - 3s 1ms/step - loss: 1.3237 -
accuracy: 0.5535
Epoch 17/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.1766 -
accuracy: 0.5985
Epoch 18/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.2648 -
accuracy: 0.5560
```

```
Epoch 19/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.1955 -
accuracy: 0.5905: 0s - loss: 1.1955 - accuracy: 0.59
Epoch 20/100
2000/2000 [==============================] - 3s 1ms/step - loss: 1.1674 -
accuracy: 0.6080
Epoch 21/100
2000/2000 [==============================] - 2s 1ms/step - loss: 1.1315 -
accuracy: 0.6115
Epoch 22/100
2000/2000 [==============================] - 2s 953us/step - loss: 1.0826 -
accuracy: 0.62051s
Epoch 23/100
2000/2000 [==============================] - 2s 787us/step - loss: 1.0949 -
accuracy: 0.6255
Epoch 24/100
2000/2000 [==============================] - 1s 734us/step - loss: 1.0493 -
accuracy: 0.6390
Epoch 25/100
2000/2000 [==============================] - 1s 728us/step - loss: 1.0591 -
accuracy: 0.6330
Epoch 26/100
2000/2000 [==============================] - 1s 730us/step - loss: 1.0335 -
accuracy: 0.6525
Epoch 27/100
2000/2000 [==============================] - 1s 746us/step - loss: 1.0075 -
accuracy: 0.6450
Epoch 28/100
2000/2000 [==============================] - 1s 730us/step - loss: 0.9678 -
accuracy: 0.6720
Epoch 29/100
2000/2000 [==============================] - 2s 918us/step - loss: 1.0129 -
accuracy: 0.6520
Epoch 30/100
2000/2000 [==============================] - 2s 901us/step - loss: 0.9687 -
accuracy: 0.6595
Epoch 31/100
2000/2000 [==============================] - 2s 955us/step - loss: 0.9172 -
accuracy: 0.6895
Epoch 32/100
2000/2000 [==============================] - 2s 832us/step - loss: 0.9052 -
accuracy: 0.6865
Epoch 33/100
2000/2000 [==============================] - 2s 913us/step - loss: 0.8806 -
accuracy: 0.6930
Epoch 34/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.8464 -
accuracy: 0.7075
```

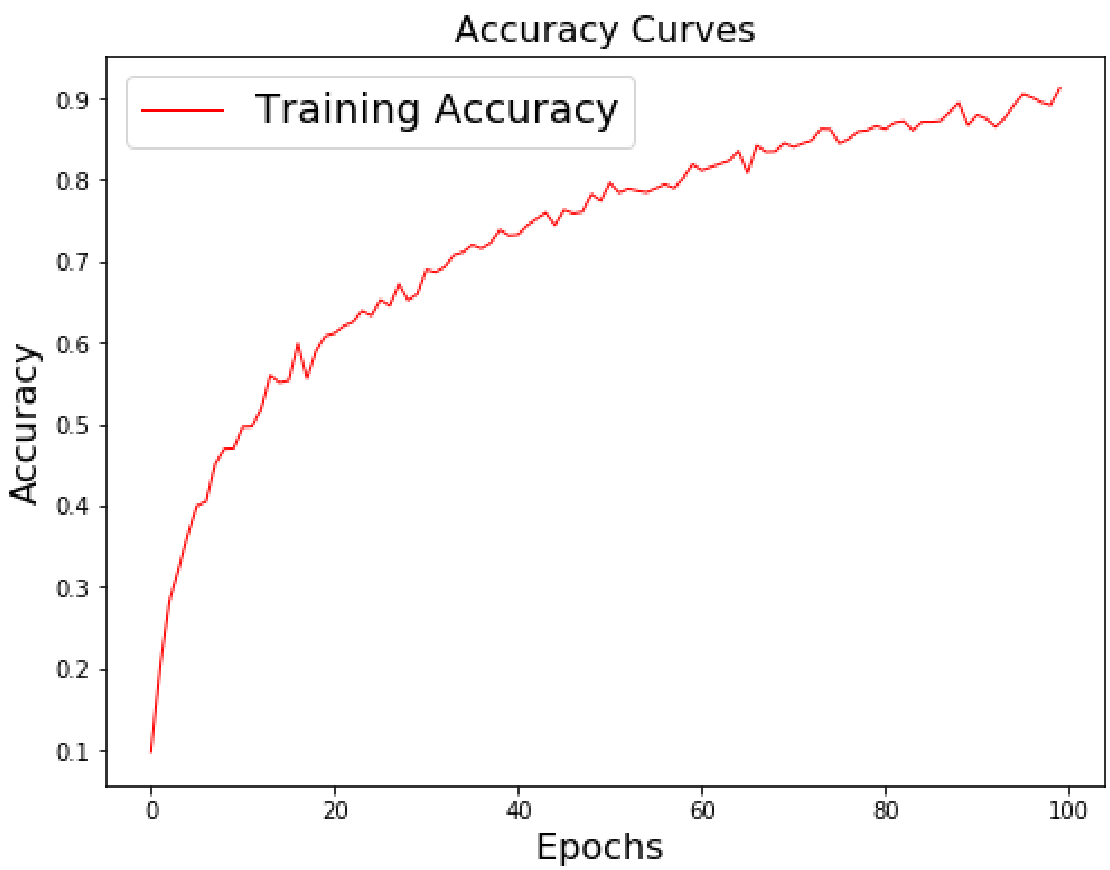
```
Epoch 35/100
2000/2000 [==============================] - 2s 857us/step - loss: 0.8443 -
accuracy: 0.7110
Epoch 36/100
2000/2000 [==============================] - 2s 953us/step - loss: 0.7887 -
accuracy: 0.7200
Epoch 37/100
2000/2000 [==============================] - 2s 750us/step - loss: 0.8209 -
accuracy: 0.7155
Epoch 38/100
2000/2000 [==============================] - 2s 841us/step - loss: 0.8098 -
accuracy: 0.7225
Epoch 39/100
2000/2000 [==============================] - 2s 810us/step - loss: 0.7952 -
accuracy: 0.7385
Epoch 40/100
2000/2000 [==============================] - 2s 750us/step - loss: 0.7986 -
accuracy: 0.7315
Epoch 41/100
2000/2000 [==============================] - 1s 726us/step - loss: 0.7655 -
accuracy: 0.7325
Epoch 42/100
2000/2000 [==============================] - 2s 883us/step - loss: 0.7523 -
accuracy: 0.7440
Epoch 43/100
2000/2000 [==============================] - 2s 844us/step - loss: 0.7324 -
accuracy: 0.7520
Epoch 44/100
2000/2000 [==============================] - 2s 879us/step - loss: 0.7128 -
accuracy: 0.75950s - loss: 0.7036 - accuracy: 0.
Epoch 45/100
2000/2000 [==============================] - 2s 847us/step - loss: 0.7191 -
accuracy: 0.7440
Epoch 46/100
2000/2000 [==============================] - 2s 851us/step - loss: 0.7354 -
accuracy: 0.7630
Epoch 47/100
2000/2000 [==============================] - 2s 865us/step - loss: 0.6836 -
accuracy: 0.7585
Epoch 48/100
2000/2000 [==============================] - 2s 833us/step - loss: 0.6988 -
accuracy: 0.7605
Epoch 49/100
2000/2000 [==============================] - 2s 881us/step - loss: 0.6303 -
accuracy: 0.7825
Epoch 50/100
2000/2000 [==============================] - 2s 892us/step - loss: 0.6538 -
accuracy: 0.7740
```

```
Epoch 51/100
2000/2000 [==============================] - 2s 827us/step - loss: 0.5883 -
accuracy: 0.7965
Epoch 52/100
2000/2000 [==============================] - 2s 849us/step - loss: 0.6166 -
accuracy: 0.7840
Epoch 53/100
2000/2000 [==============================] - 2s 981us/step - loss: 0.6008 -
accuracy: 0.7890
Epoch 54/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.6245 -
accuracy: 0.7865
Epoch 55/100
2000/2000 [==============================] - 1s 746us/step - loss: 0.6091 -
accuracy: 0.7845
Epoch 56/100
2000/2000 [==============================] - 1s 710us/step - loss: 0.6098 -
accuracy: 0.7895
Epoch 57/100
2000/2000 [==============================] - 1s 690us/step - loss: 0.6251 -
accuracy: 0.7945
Epoch 58/100
2000/2000 [==============================] - 1s 693us/step - loss: 0.6090 -
accuracy: 0.7895
Epoch 59/100
2000/2000 [==============================] - 1s 695us/step - loss: 0.5603 -
accuracy: 0.8025
Epoch 60/100
2000/2000 [==============================] - 2s 933us/step - loss: 0.5294 -
accuracy: 0.8190
Epoch 61/100
2000/2000 [==============================] - 2s 898us/step - loss: 0.5618 -
accuracy: 0.8115
Epoch 62/100
2000/2000 [==============================] - 2s 924us/step - loss: 0.5549 -
accuracy: 0.8155
Epoch 63/100
2000/2000 [==============================] - 2s 831us/step - loss: 0.5232 -
accuracy: 0.8195
Epoch 64/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.5062 -
accuracy: 0.8235
Epoch 65/100
2000/2000 [==============================] - 2s 876us/step - loss: 0.4977 -
accuracy: 0.8350
Epoch 66/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.5203 -
accuracy: 0.8085
```

```
Epoch 67/100
2000/2000 [==============================] - 2s 755us/step - loss: 0.4724 -
accuracy: 0.8420
Epoch 68/100
2000/2000 [==============================] - 2s 907us/step - loss: 0.5018 -
accuracy: 0.8340
Epoch 69/100
2000/2000 [==============================] - 2s 912us/step - loss: 0.4772 -
accuracy: 0.8345
Epoch 70/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.4488 -
accuracy: 0.8450
Epoch 71/100
2000/2000 [==============================] - 2s 952us/step - loss: 0.4805 -
accuracy: 0.8400
Epoch 72/100
2000/2000 [==============================] - 2s 876us/step - loss: 0.4579 -
accuracy: 0.8445
Epoch 73/100
2000/2000 [==============================] - 2s 834us/step - loss: 0.4257 -
accuracy: 0.8485
Epoch 74/100
2000/2000 [==============================] - 3s 2ms/step - loss: 0.4421 -
accuracy: 0.8625
Epoch 75/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.3868 -
accuracy: 0.8620
Epoch 76/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.4789 -
accuracy: 0.8445
Epoch 77/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.4521 -
accuracy: 0.8500
Epoch 78/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.4047 -
accuracy: 0.8590
Epoch 79/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.4047 -
accuracy: 0.8605
Epoch 80/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.4193 -
accuracy: 0.8660
Epoch 81/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.4102 -
accuracy: 0.8620
Epoch 82/100
2000/2000 [==============================] - 3s 2ms/step - loss: 0.3858 -
accuracy: 0.8700
```

```
Epoch 83/100
2000/2000 [==============================] - 3s 2ms/step - loss: 0.3721 -
accuracy: 0.8720
Epoch 84/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.3919 -
accuracy: 0.8605
Epoch 85/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.3791 -
accuracy: 0.8710
Epoch 86/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.3732 -
accuracy: 0.8710
Epoch 87/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3738 -
accuracy: 0.8720
Epoch 88/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.3500 -
accuracy: 0.8830
Epoch 89/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3251 -
accuracy: 0.8945
Epoch 90/100
2000/2000 [==============================] - 2s 931us/step - loss: 0.3979 -
accuracy: 0.8670
Epoch 91/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3487 -
accuracy: 0.8795
Epoch 92/100
2000/2000 [==============================] - 2s 890us/step - loss: 0.3731 -
accuracy: 0.8750
Epoch 93/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.4100 -
accuracy: 0.8650
Epoch 94/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3546 -
accuracy: 0.8755
Epoch 95/100
2000/2000 [==============================] - 2s 1ms/step - loss: 0.3288 -
accuracy: 0.8915
Epoch 96/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.2954 -
accuracy: 0.9055
Epoch 97/100
2000/2000 [==============================] - 3s 2ms/step - loss: 0.2969 -
accuracy: 0.9010
Epoch 98/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.3132 -
accuracy: 0.8950
```

```
Epoch 99/100
2000/2000 [==============================] - 3s 1ms/step - loss: 0.3239 -
accuracy: 0.8915
Epoch 100/100
2000/2000 [==============================] - 2s 960us/step - loss: 0.2765 -
accuracy: 0.9125
training  model takes 215.43 seconds
```



[44]:
```
t0 = time.time()
pred3 = model3.predict(test_data)
pred_lst3 = []
for i in range(len(pred3)):
    arr = pred3[i]
    idx = np.argwhere(arr == np.max(arr))
    pred_lst3.append(idx[0][0])
tst_labl = np.argmax(test_label_cat, axis=-1)

acc = accuracy_score(pred_lst3, tst_labl)
print("Test accuracy is %s percent" %round(acc*100,3))
print("testing model takes %s seconds" % round((time.time() - t0),3))
```

```
Test accuracy is 55.6 percent
testing model takes 0.17 seconds
```

Since second model gives the highest test accuary among three, so deceides to use this on the presentation day.

0.0.7 Run the testing data - using model2 in NN

[24]:
```
tst_root = sys.path[0]
test_dir = os.path.join(tst_root,  '../data/test_set_sec2')
test_path =  os.path.join(test_dir,  "labels_prediction.csv")
test_index = pd.read_csv(test_path)
test_pt_dir =  os.path.join(test_dir, 'points' )
```

[25]:
```
# read mat file and store coordinates in mat
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 ) ))
```

[26]:
```
mat_test = [x[[i for i in x.keys() if not i in ['__header__', '__version__',␣
↪'__globals__']][0]] for x in m_test]
```

[27]:
```
to_test_data = pairwise_dist_cal_updt(mat_test)
# choose model2
```

```
feature constructions takes 30.239994049072266 seconds
```

[28]:
```
t0 = time.time()
pred_test  = model2.predict(to_test_data)
pred_test_lst = []
for i in range(len(pred_test)):
    arr = pred_test[i]
    idx = np.argwhere(arr == np.max(arr))
    pred_test_lst.append(idx[0][0])
print("testing model takes %s seconds" % round((time.time() - t0),3))
```

```
testing model takes 0.315 seconds
```

[29]:
```
final_pred = pd.DataFrame([1+x for x in pred_test_lst]).to_csv("../output/
↪final_model.csv",index=False)
```

[ ]:
```
```