

Lab09a_pca_NN_CNN_partial

April 30, 2018

1 Lab 9a: PCA for Face Recognition

Following the demo for this unit, we will explore further the use of PCA for feature dimension reduction for classification. We will use a 2-layer neural net on the PCA coefficients. We will practice optimizing the classification parameters (the number of PCA components and the number of hidden nodes in the NN classifier). We will furthermore compare this approach with using convolutional neural net on raw images.

Through the lab, you will learn to:

- Perform PCA on the a face dataset to find the PC components
- Evaluate the effect of using different number of principle components for data representation and classification.
- Optimize the number of PC coefficients and classifier parameters together to maximize classification accuracy.
- Understand the impact of training data size on the feature and classification method selection.

```
In [1]: import numpy as np
import matplotlib
import matplotlib.pyplot as plt
```

```
In [3]: # Import the flw_people dataset.
# Select only those people with at least 100 instances
# Reduce the face image size by 0.4

# TO DO
from sklearn.datasets import fetch_lfw_people
lfw_people = fetch_lfw_people(min_faces_per_person=100, resize=0.4)
```

```
Downloading LFW metadata: https://ndownloader.figshare.com/files/5976012
Downloading LFW metadata: https://ndownloader.figshare.com/files/5976009
Downloading LFW metadata: https://ndownloader.figshare.com/files/5976006
Downloading LFW data (~200MB): https://ndownloader.figshare.com/files/5976015
```

```
In [4]: # Save the face images in a datamatrix X and the labels and corresponding names in a d
# Furthermore, determine the number of samples and the image size
```

```

# Determine the number of different faces (number of classes)

# TO DO
n_samples, h, w = lfw_people.images.shape
npix = h*w
# Data in 2D form
X = lfw_people.data
n_features = X.shape[1]
# Labels of images
y = lfw_people.target
target_names = lfw_people.target_names
n_classes = target_names.shape[0]
print("Image size = {0:d} x {1:d} = {2:d} pixels".format(h,w,npix))
print("Number faces = {0:d}".format(n_samples))
print("Number classes = {0:d}".format(n_classes))

```

Image size = 50 x 37 = 1850 pixels

Number faces = 1140

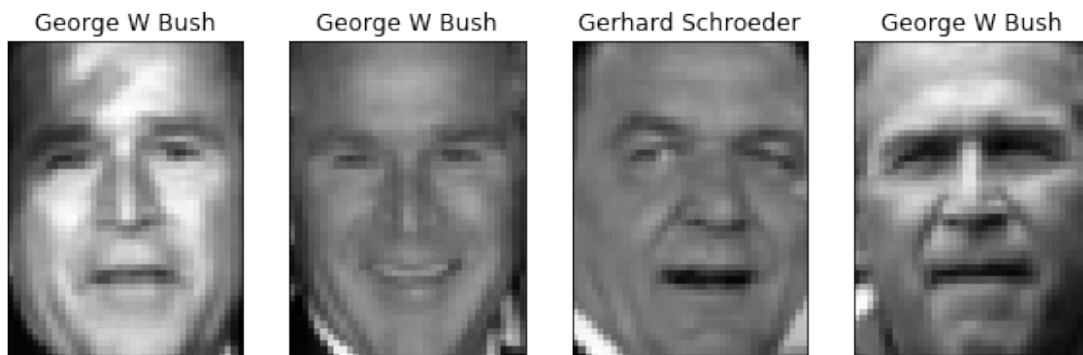
Number classes = 5

In [6]: # Plot some sample images to make sure your data load is correct

```

def plt_face(x):
    h = 50
    w = 37
    plt.imshow(x.reshape((h, w)), cmap=plt.cm.gray)
    plt.xticks([])
    plt.yticks([])
I = np.random.permutation(n_samples)
plt.figure(figsize=(10,20))
nplt = 4;
for i in range(nplt):
    ind = I[i]
    plt.subplot(1,nplt,i+1)
    plt_face(X[ind])
    plt.title(target_names[y[ind]])

```



```
In [7]: # Split the data into a training set and test set with 50% data for training.
        # Use "stratify" option to make sure the training data and test data have same
        # proportion of images from different faces
        # print the number of samples in the training data

        # TO DO
        from sklearn.model_selection import train_test_split
        # split into a training and testing set
        X_train, X_test, y_train, y_test = train_test_split(
            X, y, stratify=y, test_size=0.5)

        n_samples_train = X_train.shape[0]
        print("Number faces in training data = {0:d}".format(n_samples_train))
```

Number faces in training data = 570

```
In [8]: # Perform PCA on the training data to derive the principle components (PCs) and the PCA
        # You can directly use the PCA class in PCA package or use SVD.
        # Remember that you need to remove the mean from the data first
        # Also you should rescale the PCs so that the PCA coefficients all have unit variance
        # Determine the total number of PCs

        # TO DO
        from sklearn.model_selection import GridSearchCV
        from sklearn.svm import SVC
        n_samples, _ = X_train.shape
        Xtr_mean = np.mean(X_train, 0)
        Xtr = X_train - Xtr_mean[None, :]
        Utr, Str, Vtr = np.linalg.svd(Xtr, full_matrices=False)
        print("The total number of PCs is %d." % Vtr.shape[0])
```

The total number of PCs is 570.

First let us construct a 2-layer neural net classifier that uses $n_{pc}=100$ PCA coefficients to classify the faces. Set up your training and testing data to contain n_{pc} PCA coefficients using the previously determined principle components. You should directly use matrix multiplication (i.e. projecting original data to the first 100 principle components you found previously) to find the coefficients rather than using the `pca.transform()` method.

```
In [10]: # TO DO
         npc = 100
         eigenface = Vtr[:npc, :]
         Xtr_pca = Xtr.dot(eigenface.T)
         Xtr_pca_s = Xtr_pca / Str[None, :npc] * np.sqrt(n_samples)
```

```

Xts = X_test - Xtr_mean[None,:]
Xts_pca = Xts.dot(eigenface.T)
Xts_pca_s = Xts_pca / Str[None,:npc] * np.sqrt(n_samples)

```

Now set up and compile a NN model with number of hidden nodes nnode=100 and a output layer, and then fit the model to the training data. Use 'relu' for the activation for the hidden layer and use 'softmax' for the output layer. Using sparse_categorical_crossentropy for the loss. Use accuracy as the metrics. You can choose to do a small number of epochs (=10) with batch size =100. Determine the accuracy on the validation set.

```

In [15]: # TO DO
import keras
from keras.models import Model, Sequential
from keras.layers import Dense, Activation
from keras.layers import Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
import keras.backend as K
K.clear_session()
nin = Xtr_pca.shape[1] # dimension of input data
nh = 100 # number of hidden units
nout = int(np.max(y_train)+1) # number of outputs
model = Sequential()
model.add(Dense(nh, input_shape=(nin,), activation='relu', name='hidden'))
model.add(Dense(nout, activation='softmax', name='output'))
model.summary()

```

Layer (type)	Output Shape	Param #
hidden (Dense)	(None, 100)	10100
output (Dense)	(None, 5)	505

Total params: 10,605
 Trainable params: 10,605
 Non-trainable params: 0

Now try to identify the best number of PCs and the best number of hidden nodes in the NN classifier that can achieve the highest validation accuracy. You can set the range of PCs and nubmer of hidden nodes as below.

```

nnodes = [50,100,150,200, 250], npcs = [50,100,150,200]

```

```

In [17]: # Set up an array to store accuracy for different nnode and npcs
# TO DO
from keras import optimizers
opt = optimizers.Adam(lr=0.01, beta_1=0.9, beta_2=0.999, epsilon=1e-08, decay=0.0)
model.compile(optimizer=opt,

```

```

        loss='sparse_categorical_crossentropy',
        metrics=['accuracy'])
hist = model.fit(Xtr_pca_s, y_train, epochs=10, batch_size=100)

nnodes = [50,100,150,200,250]
npcs = [50,100,150,200]

Epoch 1/10
570/570 [=====] - 0s 376us/step - loss: 0.0086 - acc: 0.9982
Epoch 2/10
570/570 [=====] - 0s 35us/step - loss: 0.0031 - acc: 1.0000
Epoch 3/10
570/570 [=====] - 0s 35us/step - loss: 0.0014 - acc: 1.0000
Epoch 4/10
570/570 [=====] - 0s 38us/step - loss: 8.5248e-04 - acc: 1.0000
Epoch 5/10
570/570 [=====] - 0s 34us/step - loss: 4.0785e-04 - acc: 1.0000
Epoch 6/10
570/570 [=====] - 0s 30us/step - loss: 2.4188e-04 - acc: 1.0000
Epoch 7/10
570/570 [=====] - 0s 30us/step - loss: 1.5982e-04 - acc: 1.0000
Epoch 8/10
570/570 [=====] - 0s 36us/step - loss: 1.1410e-04 - acc: 1.0000
Epoch 9/10
570/570 [=====] - 0s 26us/step - loss: 8.8744e-05 - acc: 1.0000
Epoch 10/10
570/570 [=====] - 0s 34us/step - loss: 7.2105e-05 - acc: 1.0000

```

```

In [19]: # Loop through the combinations to find the accuracy for each combination
         # For each possible combination of `nnode` and `npc`, set up and fit the model
         # using features containing only coefficents corresponding to npc coefficients.

```

```

# TO DO
result = np.zeros((len(npcs),len(nnodes)))
loss_hist = []
train_acc_hist = []
val_acc_hist = []
for i,npc in enumerate(npcs):
    for j,nnode in enumerate(nnodes):
        K.clear_session()
        eigenface = Vtr[:,npc,:]
        Xtr_pca = X_train.dot(eigenface.T)
        Xtr_pca_s = Xtr_pca / Str[None,:npc] * np.sqrt(n_samples)
        Xts = X_test - Xtr_mean[None,:]
        Xts_pca = Xts.dot(eigenface.T)
        Xts_pca_s = Xts_pca / Str[None,:npc] * np.sqrt(n_samples)
        nin = Xtr_pca.shape[1] # dimension of input data

```

```

nh = nnode # number of hidden units
nout = int(np.max(y_train)+1)
model = Sequential()
model.add(Dense(nh, input_shape=(nin,), activation='relu', name='hidden'))
model.add(Dense(nout, activation='softmax', name='output'))
opt = optimizers.Adam(lr=0.01, beta_1=0.9, beta_2=0.999, epsilon=1e-08, decay=0.01)
model.compile(optimizer=opt, loss='sparse_categorical_crossentropy', metrics=['accuracy'])
hist = model.fit(Xtr_pca_s, y_train, epochs=10, batch_size=100,
                 validation_data=(Xts_pca_s, y_test), shuffle=True)
result[i][j] = hist.history['val_acc'][-1]

```

Train on 570 samples, validate on 570 samples

```

Epoch 1/10
570/570 [=====] - 0s 363us/step - loss: 1.5764 - acc: 0.4175 - val_loss: 1.5764
Epoch 2/10
570/570 [=====] - 0s 36us/step - loss: 1.0512 - acc: 0.6263 - val_loss: 1.0512
Epoch 3/10
570/570 [=====] - 0s 44us/step - loss: 0.7170 - acc: 0.7877 - val_loss: 0.7170
Epoch 4/10
570/570 [=====] - 0s 45us/step - loss: 0.5308 - acc: 0.8491 - val_loss: 0.5308
Epoch 5/10
570/570 [=====] - 0s 43us/step - loss: 0.3848 - acc: 0.8877 - val_loss: 0.3848
Epoch 6/10
570/570 [=====] - 0s 34us/step - loss: 0.3033 - acc: 0.9158 - val_loss: 0.3033
Epoch 7/10
570/570 [=====] - 0s 39us/step - loss: 0.2402 - acc: 0.9368 - val_loss: 0.2402
Epoch 8/10
570/570 [=====] - 0s 38us/step - loss: 0.2031 - acc: 0.9474 - val_loss: 0.2031
Epoch 9/10
570/570 [=====] - 0s 40us/step - loss: 0.1678 - acc: 0.9614 - val_loss: 0.1678
Epoch 10/10
570/570 [=====] - 0s 36us/step - loss: 0.1440 - acc: 0.9702 - val_loss: 0.1440

```

Train on 570 samples, validate on 570 samples

```

Epoch 1/10
570/570 [=====] - 0s 376us/step - loss: 1.5896 - acc: 0.4053 - val_loss: 1.5896
Epoch 2/10
570/570 [=====] - 0s 38us/step - loss: 0.8556 - acc: 0.6702 - val_loss: 0.8556
Epoch 3/10
570/570 [=====] - 0s 53us/step - loss: 0.5079 - acc: 0.8737 - val_loss: 0.5079
Epoch 4/10
570/570 [=====] - 0s 50us/step - loss: 0.3494 - acc: 0.9035 - val_loss: 0.3494
Epoch 5/10
570/570 [=====] - 0s 50us/step - loss: 0.2530 - acc: 0.9281 - val_loss: 0.2530
Epoch 6/10
570/570 [=====] - 0s 43us/step - loss: 0.1989 - acc: 0.9456 - val_loss: 0.1989
Epoch 7/10
570/570 [=====] - 0s 48us/step - loss: 0.1576 - acc: 0.9632 - val_loss: 0.1576
Epoch 8/10

```

```

570/570 [=====] - 0s 39us/step - loss: 0.1217 - acc: 0.9789 - val_loss: 0.1217
Epoch 9/10
570/570 [=====] - 0s 43us/step - loss: 0.0939 - acc: 0.9895 - val_loss: 0.0939
Epoch 10/10
570/570 [=====] - 0s 48us/step - loss: 0.0751 - acc: 0.9912 - val_loss: 0.0751
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 375us/step - loss: 1.6693 - acc: 0.4368 - val_loss: 1.6693
Epoch 2/10
570/570 [=====] - 0s 41us/step - loss: 0.8089 - acc: 0.7228 - val_loss: 0.8089
Epoch 3/10
570/570 [=====] - 0s 50us/step - loss: 0.5032 - acc: 0.8456 - val_loss: 0.5032
Epoch 4/10
570/570 [=====] - 0s 44us/step - loss: 0.3448 - acc: 0.8895 - val_loss: 0.3448
Epoch 5/10
570/570 [=====] - 0s 49us/step - loss: 0.2658 - acc: 0.9228 - val_loss: 0.2658
Epoch 6/10
570/570 [=====] - 0s 48us/step - loss: 0.2091 - acc: 0.9421 - val_loss: 0.2091
Epoch 7/10
570/570 [=====] - 0s 109us/step - loss: 0.1496 - acc: 0.9737 - val_loss: 0.1496
Epoch 8/10
570/570 [=====] - 0s 90us/step - loss: 0.1149 - acc: 0.9807 - val_loss: 0.1149
Epoch 9/10
570/570 [=====] - 0s 56us/step - loss: 0.0932 - acc: 0.9895 - val_loss: 0.0932
Epoch 10/10
570/570 [=====] - 0s 58us/step - loss: 0.0725 - acc: 0.9860 - val_loss: 0.0725
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 497us/step - loss: 1.8334 - acc: 0.3947 - val_loss: 1.8334
Epoch 2/10
570/570 [=====] - 0s 53us/step - loss: 0.9750 - acc: 0.6632 - val_loss: 0.9750
Epoch 3/10
570/570 [=====] - 0s 44us/step - loss: 0.5965 - acc: 0.8053 - val_loss: 0.5965
Epoch 4/10
570/570 [=====] - 0s 64us/step - loss: 0.3893 - acc: 0.8772 - val_loss: 0.3893
Epoch 5/10
570/570 [=====] - 0s 51us/step - loss: 0.2602 - acc: 0.9246 - val_loss: 0.2602
Epoch 6/10
570/570 [=====] - 0s 57us/step - loss: 0.2054 - acc: 0.9474 - val_loss: 0.2054
Epoch 7/10
570/570 [=====] - 0s 48us/step - loss: 0.1513 - acc: 0.9596 - val_loss: 0.1513
Epoch 8/10
570/570 [=====] - 0s 59us/step - loss: 0.1092 - acc: 0.9825 - val_loss: 0.1092
Epoch 9/10
570/570 [=====] - 0s 47us/step - loss: 0.0822 - acc: 0.9895 - val_loss: 0.0822
Epoch 10/10
570/570 [=====] - 0s 57us/step - loss: 0.0610 - acc: 0.9912 - val_loss: 0.0610
Train on 570 samples, validate on 570 samples

```

```

Epoch 1/10
570/570 [=====] - 0s 485us/step - loss: 1.4528 - acc: 0.4772 - val_loss: 1.4528
Epoch 2/10
570/570 [=====] - 0s 54us/step - loss: 0.7021 - acc: 0.7421 - val_loss: 0.7021
Epoch 3/10
570/570 [=====] - 0s 46us/step - loss: 0.4844 - acc: 0.8544 - val_loss: 0.4844
Epoch 4/10
570/570 [=====] - 0s 49us/step - loss: 0.2950 - acc: 0.9035 - val_loss: 0.2950
Epoch 5/10
570/570 [=====] - 0s 49us/step - loss: 0.2129 - acc: 0.9404 - val_loss: 0.2129
Epoch 6/10
570/570 [=====] - 0s 45us/step - loss: 0.1460 - acc: 0.9719 - val_loss: 0.1460
Epoch 7/10
570/570 [=====] - 0s 46us/step - loss: 0.1151 - acc: 0.9807 - val_loss: 0.1151
Epoch 8/10
570/570 [=====] - 0s 49us/step - loss: 0.0766 - acc: 0.9895 - val_loss: 0.0766
Epoch 9/10
570/570 [=====] - 0s 54us/step - loss: 0.0563 - acc: 0.9947 - val_loss: 0.0563
Epoch 10/10
570/570 [=====] - 0s 91us/step - loss: 0.0409 - acc: 0.9965 - val_loss: 0.0409
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 420us/step - loss: 1.6058 - acc: 0.3544 - val_loss: 1.6058
Epoch 2/10
570/570 [=====] - 0s 62us/step - loss: 1.1019 - acc: 0.6175 - val_loss: 1.1019
Epoch 3/10
570/570 [=====] - 0s 50us/step - loss: 0.7890 - acc: 0.7246 - val_loss: 0.7890
Epoch 4/10
570/570 [=====] - 0s 49us/step - loss: 0.4885 - acc: 0.8649 - val_loss: 0.4885
Epoch 5/10
570/570 [=====] - 0s 51us/step - loss: 0.3013 - acc: 0.9333 - val_loss: 0.3013
Epoch 6/10
570/570 [=====] - 0s 57us/step - loss: 0.2145 - acc: 0.9474 - val_loss: 0.2145
Epoch 7/10
570/570 [=====] - 0s 56us/step - loss: 0.1478 - acc: 0.9789 - val_loss: 0.1478
Epoch 8/10
570/570 [=====] - 0s 47us/step - loss: 0.0956 - acc: 0.9842 - val_loss: 0.0956
Epoch 9/10
570/570 [=====] - 0s 37us/step - loss: 0.0646 - acc: 0.9947 - val_loss: 0.0646
Epoch 10/10
570/570 [=====] - 0s 44us/step - loss: 0.0440 - acc: 0.9947 - val_loss: 0.0440
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 364us/step - loss: 1.5875 - acc: 0.4667 - val_loss: 1.5875
Epoch 2/10
570/570 [=====] - 0s 38us/step - loss: 0.7907 - acc: 0.7544 - val_loss: 0.7907
Epoch 3/10
570/570 [=====] - 0s 50us/step - loss: 0.3971 - acc: 0.8982 - val_loss: 0.3971

```



```

Epoch 4/10
570/570 [=====] - 0s 55us/step - loss: 0.2126 - acc: 0.9596 - val_loss: 0.2126
Epoch 5/10
570/570 [=====] - 0s 75us/step - loss: 0.1167 - acc: 0.9877 - val_loss: 0.1167
Epoch 6/10
570/570 [=====] - 0s 63us/step - loss: 0.0722 - acc: 0.9912 - val_loss: 0.0722
Epoch 7/10
570/570 [=====] - 0s 48us/step - loss: 0.0412 - acc: 0.9965 - val_loss: 0.0412
Epoch 8/10
570/570 [=====] - 0s 79us/step - loss: 0.0254 - acc: 1.0000 - val_loss: 0.0254
Epoch 9/10
570/570 [=====] - 0s 55us/step - loss: 0.0178 - acc: 1.0000 - val_loss: 0.0178
Epoch 10/10
570/570 [=====] - 0s 55us/step - loss: 0.0126 - acc: 1.0000 - val_loss: 0.0126
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 432us/step - loss: 1.7754 - acc: 0.4246 - val_loss: 1.7754
Epoch 2/10
570/570 [=====] - 0s 48us/step - loss: 0.8183 - acc: 0.7140 - val_loss: 0.8183
Epoch 3/10
570/570 [=====] - 0s 56us/step - loss: 0.4437 - acc: 0.9105 - val_loss: 0.4437
Epoch 4/10
570/570 [=====] - 0s 59us/step - loss: 0.2544 - acc: 0.9316 - val_loss: 0.2544
Epoch 5/10
570/570 [=====] - 0s 59us/step - loss: 0.1400 - acc: 0.9719 - val_loss: 0.1400
Epoch 6/10
570/570 [=====] - 0s 62us/step - loss: 0.0783 - acc: 0.9877 - val_loss: 0.0783
Epoch 7/10
570/570 [=====] - 0s 51us/step - loss: 0.0427 - acc: 0.9965 - val_loss: 0.0427
Epoch 8/10
570/570 [=====] - 0s 47us/step - loss: 0.0278 - acc: 1.0000 - val_loss: 0.0278
Epoch 9/10
570/570 [=====] - 0s 47us/step - loss: 0.0180 - acc: 1.0000 - val_loss: 0.0180
Epoch 10/10
570/570 [=====] - 0s 51us/step - loss: 0.0125 - acc: 1.0000 - val_loss: 0.0125
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 398us/step - loss: 1.8298 - acc: 0.4368 - val_loss: 1.8298
Epoch 2/10
570/570 [=====] - 0s 39us/step - loss: 0.8003 - acc: 0.8053 - val_loss: 0.8003
Epoch 3/10
570/570 [=====] - 0s 51us/step - loss: 0.4108 - acc: 0.9018 - val_loss: 0.4108
Epoch 4/10
570/570 [=====] - 0s 102us/step - loss: 0.2311 - acc: 0.9298 - val_loss: 0.2311
Epoch 5/10
570/570 [=====] - 0s 71us/step - loss: 0.1173 - acc: 0.9772 - val_loss: 0.1173
Epoch 6/10
570/570 [=====] - 0s 64us/step - loss: 0.0643 - acc: 0.9930 - val_loss: 0.0643

```

```

Epoch 7/10
570/570 [=====] - 0s 54us/step - loss: 0.0347 - acc: 1.0000 - val_loss: 0.0347
Epoch 8/10
570/570 [=====] - 0s 57us/step - loss: 0.0206 - acc: 1.0000 - val_loss: 0.0206
Epoch 9/10
570/570 [=====] - 0s 51us/step - loss: 0.0132 - acc: 1.0000 - val_loss: 0.0132
Epoch 10/10
570/570 [=====] - 0s 62us/step - loss: 0.0095 - acc: 1.0000 - val_loss: 0.0095
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 483us/step - loss: 2.2324 - acc: 0.3561 - val_loss: 2.2324
Epoch 2/10
570/570 [=====] - 0s 46us/step - loss: 0.8449 - acc: 0.6772 - val_loss: 0.8449
Epoch 3/10
570/570 [=====] - 0s 53us/step - loss: 0.4990 - acc: 0.8421 - val_loss: 0.4990
Epoch 4/10
570/570 [=====] - 0s 48us/step - loss: 0.2864 - acc: 0.9439 - val_loss: 0.2864
Epoch 5/10
570/570 [=====] - 0s 52us/step - loss: 0.1511 - acc: 0.9632 - val_loss: 0.1511
Epoch 6/10
570/570 [=====] - 0s 59us/step - loss: 0.0868 - acc: 0.9860 - val_loss: 0.0868
Epoch 7/10
570/570 [=====] - 0s 51us/step - loss: 0.0480 - acc: 0.9965 - val_loss: 0.0480
Epoch 8/10
570/570 [=====] - 0s 62us/step - loss: 0.0246 - acc: 1.0000 - val_loss: 0.0246
Epoch 9/10
570/570 [=====] - 0s 58us/step - loss: 0.0163 - acc: 1.0000 - val_loss: 0.0163
Epoch 10/10
570/570 [=====] - 0s 82us/step - loss: 0.0099 - acc: 1.0000 - val_loss: 0.0099
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 508us/step - loss: 1.9450 - acc: 0.3421 - val_loss: 1.9450
Epoch 2/10
570/570 [=====] - 0s 46us/step - loss: 1.1963 - acc: 0.5702 - val_loss: 1.1963
Epoch 3/10
570/570 [=====] - 0s 51us/step - loss: 0.9007 - acc: 0.7070 - val_loss: 0.9007
Epoch 4/10
570/570 [=====] - 0s 35us/step - loss: 0.5927 - acc: 0.8158 - val_loss: 0.5927
Epoch 5/10
570/570 [=====] - 0s 42us/step - loss: 0.3642 - acc: 0.9316 - val_loss: 0.3642
Epoch 6/10
570/570 [=====] - 0s 40us/step - loss: 0.2084 - acc: 0.9596 - val_loss: 0.2084
Epoch 7/10
570/570 [=====] - 0s 35us/step - loss: 0.1134 - acc: 0.9895 - val_loss: 0.1134
Epoch 8/10
570/570 [=====] - 0s 38us/step - loss: 0.0670 - acc: 0.9947 - val_loss: 0.0670
Epoch 9/10
570/570 [=====] - 0s 37us/step - loss: 0.0398 - acc: 1.0000 - val_loss: 0.0398

```

```

Epoch 10/10
570/570 [=====] - 0s 46us/step - loss: 0.0255 - acc: 1.0000 - val_loss: 0.0255
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 365us/step - loss: 1.9493 - acc: 0.3807 - val_loss: 1.9493
Epoch 2/10
570/570 [=====] - 0s 39us/step - loss: 0.9423 - acc: 0.6825 - val_loss: 0.9423
Epoch 3/10
570/570 [=====] - 0s 43us/step - loss: 0.5678 - acc: 0.8421 - val_loss: 0.5678
Epoch 4/10
570/570 [=====] - 0s 41us/step - loss: 0.2989 - acc: 0.9246 - val_loss: 0.2989
Epoch 5/10
570/570 [=====] - 0s 38us/step - loss: 0.1483 - acc: 0.9807 - val_loss: 0.1483
Epoch 6/10
570/570 [=====] - 0s 40us/step - loss: 0.0759 - acc: 0.9930 - val_loss: 0.0759
Epoch 7/10
570/570 [=====] - 0s 36us/step - loss: 0.0407 - acc: 0.9982 - val_loss: 0.0407
Epoch 8/10
570/570 [=====] - 0s 45us/step - loss: 0.0248 - acc: 1.0000 - val_loss: 0.0248
Epoch 9/10
570/570 [=====] - 0s 44us/step - loss: 0.0159 - acc: 1.0000 - val_loss: 0.0159
Epoch 10/10
570/570 [=====] - 0s 45us/step - loss: 0.0113 - acc: 1.0000 - val_loss: 0.0113
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 387us/step - loss: 2.4614 - acc: 0.3965 - val_loss: 2.4614
Epoch 2/10
570/570 [=====] - 0s 41us/step - loss: 1.0770 - acc: 0.6895 - val_loss: 1.0770
Epoch 3/10
570/570 [=====] - 0s 49us/step - loss: 0.6413 - acc: 0.8351 - val_loss: 0.6413
Epoch 4/10
570/570 [=====] - 0s 49us/step - loss: 0.4009 - acc: 0.9070 - val_loss: 0.4009
Epoch 5/10
570/570 [=====] - 0s 48us/step - loss: 0.2585 - acc: 0.9193 - val_loss: 0.2585
Epoch 6/10
570/570 [=====] - 0s 49us/step - loss: 0.1391 - acc: 0.9737 - val_loss: 0.1391
Epoch 7/10
570/570 [=====] - 0s 48us/step - loss: 0.0753 - acc: 0.9947 - val_loss: 0.0753
Epoch 8/10
570/570 [=====] - 0s 56us/step - loss: 0.0434 - acc: 0.9947 - val_loss: 0.0434
Epoch 9/10
570/570 [=====] - 0s 52us/step - loss: 0.0240 - acc: 1.0000 - val_loss: 0.0240
Epoch 10/10
570/570 [=====] - 0s 49us/step - loss: 0.0157 - acc: 1.0000 - val_loss: 0.0157
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 387us/step - loss: 2.7022 - acc: 0.3509 - val_loss: 2.7022
Epoch 2/10

```

```

570/570 [=====] - 0s 52us/step - loss: 1.0771 - acc: 0.6158 - val_loss: 1.0771
Epoch 3/10
570/570 [=====] - 0s 54us/step - loss: 0.6682 - acc: 0.7912 - val_loss: 0.6682
Epoch 4/10
570/570 [=====] - 0s 49us/step - loss: 0.4051 - acc: 0.8895 - val_loss: 0.4051
Epoch 5/10
570/570 [=====] - 0s 56us/step - loss: 0.2041 - acc: 0.9667 - val_loss: 0.2041
Epoch 6/10
570/570 [=====] - 0s 54us/step - loss: 0.1056 - acc: 0.9860 - val_loss: 0.1056
Epoch 7/10
570/570 [=====] - 0s 55us/step - loss: 0.0492 - acc: 0.9982 - val_loss: 0.0492
Epoch 8/10
570/570 [=====] - 0s 50us/step - loss: 0.0290 - acc: 0.9982 - val_loss: 0.0290
Epoch 9/10
570/570 [=====] - 0s 55us/step - loss: 0.0159 - acc: 1.0000 - val_loss: 0.0159
Epoch 10/10
570/570 [=====] - 0s 57us/step - loss: 0.0087 - acc: 1.0000 - val_loss: 0.0087
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 446us/step - loss: 2.8959 - acc: 0.3018 - val_loss: 2.8959
Epoch 2/10
570/570 [=====] - 0s 59us/step - loss: 1.0104 - acc: 0.6053 - val_loss: 1.0104
Epoch 3/10
570/570 [=====] - 0s 56us/step - loss: 0.7345 - acc: 0.7351 - val_loss: 0.7345
Epoch 4/10
570/570 [=====] - 0s 52us/step - loss: 0.4059 - acc: 0.9333 - val_loss: 0.4059
Epoch 5/10
570/570 [=====] - 0s 50us/step - loss: 0.1962 - acc: 0.9614 - val_loss: 0.1962
Epoch 6/10
570/570 [=====] - 0s 54us/step - loss: 0.0926 - acc: 0.9860 - val_loss: 0.0926
Epoch 7/10
570/570 [=====] - 0s 55us/step - loss: 0.0443 - acc: 0.9930 - val_loss: 0.0443
Epoch 8/10
570/570 [=====] - 0s 59us/step - loss: 0.0213 - acc: 1.0000 - val_loss: 0.0213
Epoch 9/10
570/570 [=====] - 0s 63us/step - loss: 0.0123 - acc: 1.0000 - val_loss: 0.0123
Epoch 10/10
570/570 [=====] - 0s 58us/step - loss: 0.0077 - acc: 1.0000 - val_loss: 0.0077
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 368us/step - loss: 1.8770 - acc: 0.3649 - val_loss: 1.8770
Epoch 2/10
570/570 [=====] - 0s 35us/step - loss: 1.0651 - acc: 0.5737 - val_loss: 1.0651
Epoch 3/10
570/570 [=====] - 0s 38us/step - loss: 0.6934 - acc: 0.8070 - val_loss: 0.6934
Epoch 4/10
570/570 [=====] - 0s 46us/step - loss: 0.4012 - acc: 0.9000 - val_loss: 0.4012
Epoch 5/10

```

```

570/570 [=====] - 0s 47us/step - loss: 0.2100 - acc: 0.9684 - val_loss: 0.2100
Epoch 6/10
570/570 [=====] - 0s 39us/step - loss: 0.1101 - acc: 0.9895 - val_loss: 0.1101
Epoch 7/10
570/570 [=====] - 0s 44us/step - loss: 0.0539 - acc: 0.9982 - val_loss: 0.0539
Epoch 8/10
570/570 [=====] - 0s 45us/step - loss: 0.0286 - acc: 1.0000 - val_loss: 0.0286
Epoch 9/10
570/570 [=====] - 0s 37us/step - loss: 0.0177 - acc: 1.0000 - val_loss: 0.0177
Epoch 10/10
570/570 [=====] - 0s 46us/step - loss: 0.0119 - acc: 1.0000 - val_loss: 0.0119
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 638us/step - loss: 2.1762 - acc: 0.3158 - val_loss: 2.1762
Epoch 2/10
570/570 [=====] - 0s 46us/step - loss: 1.0503 - acc: 0.6070 - val_loss: 1.0503
Epoch 3/10
570/570 [=====] - 0s 48us/step - loss: 0.6806 - acc: 0.8088 - val_loss: 0.6806
Epoch 4/10
570/570 [=====] - 0s 50us/step - loss: 0.3719 - acc: 0.9088 - val_loss: 0.3719
Epoch 5/10
570/570 [=====] - 0s 47us/step - loss: 0.1836 - acc: 0.9719 - val_loss: 0.1836
Epoch 6/10
570/570 [=====] - 0s 46us/step - loss: 0.0773 - acc: 0.9947 - val_loss: 0.0773
Epoch 7/10
570/570 [=====] - 0s 43us/step - loss: 0.0377 - acc: 1.0000 - val_loss: 0.0377
Epoch 8/10
570/570 [=====] - 0s 41us/step - loss: 0.0203 - acc: 1.0000 - val_loss: 0.0203
Epoch 9/10
570/570 [=====] - 0s 44us/step - loss: 0.0113 - acc: 1.0000 - val_loss: 0.0113
Epoch 10/10
570/570 [=====] - 0s 46us/step - loss: 0.0085 - acc: 1.0000 - val_loss: 0.0085
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 432us/step - loss: 1.9143 - acc: 0.3789 - val_loss: 1.9143
Epoch 2/10
570/570 [=====] - 0s 60us/step - loss: 0.8178 - acc: 0.7175 - val_loss: 0.8178
Epoch 3/10
570/570 [=====] - 0s 54us/step - loss: 0.4119 - acc: 0.9351 - val_loss: 0.4119
Epoch 4/10
570/570 [=====] - 0s 54us/step - loss: 0.1854 - acc: 0.9561 - val_loss: 0.1854
Epoch 5/10
570/570 [=====] - 0s 41us/step - loss: 0.0694 - acc: 0.9930 - val_loss: 0.0694
Epoch 6/10
570/570 [=====] - 0s 47us/step - loss: 0.0399 - acc: 0.9930 - val_loss: 0.0399
Epoch 7/10
570/570 [=====] - 0s 47us/step - loss: 0.0180 - acc: 1.0000 - val_loss: 0.0180
Epoch 8/10

```

```

570/570 [=====] - 0s 42us/step - loss: 0.0110 - acc: 1.0000 - val_loss: 0.0110
Epoch 9/10
570/570 [=====] - 0s 147us/step - loss: 0.0059 - acc: 1.0000 - val_loss: 0.0059
Epoch 10/10
570/570 [=====] - 0s 53us/step - loss: 0.0046 - acc: 1.0000 - val_loss: 0.0046
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 477us/step - loss: 2.5762 - acc: 0.3632 - val_loss: 2.5762
Epoch 2/10
570/570 [=====] - 0s 46us/step - loss: 1.0014 - acc: 0.6456 - val_loss: 1.0014
Epoch 3/10
570/570 [=====] - 0s 58us/step - loss: 0.6294 - acc: 0.8123 - val_loss: 0.6294
Epoch 4/10
570/570 [=====] - 0s 54us/step - loss: 0.3459 - acc: 0.9035 - val_loss: 0.3459
Epoch 5/10
570/570 [=====] - 0s 56us/step - loss: 0.1493 - acc: 0.9807 - val_loss: 0.1493
Epoch 6/10
570/570 [=====] - 0s 55us/step - loss: 0.0600 - acc: 0.9965 - val_loss: 0.0600
Epoch 7/10
570/570 [=====] - 0s 55us/step - loss: 0.0258 - acc: 1.0000 - val_loss: 0.0258
Epoch 8/10
570/570 [=====] - 0s 53us/step - loss: 0.0112 - acc: 1.0000 - val_loss: 0.0112
Epoch 9/10
570/570 [=====] - 0s 48us/step - loss: 0.0067 - acc: 1.0000 - val_loss: 0.0067
Epoch 10/10
570/570 [=====] - 0s 50us/step - loss: 0.0042 - acc: 1.0000 - val_loss: 0.0042
Train on 570 samples, validate on 570 samples
Epoch 1/10
570/570 [=====] - 0s 393us/step - loss: 2.7537 - acc: 0.3596 - val_loss: 2.7537
Epoch 2/10
570/570 [=====] - 0s 48us/step - loss: 1.0996 - acc: 0.6246 - val_loss: 1.0996
Epoch 3/10
570/570 [=====] - 0s 53us/step - loss: 0.6900 - acc: 0.7719 - val_loss: 0.6900
Epoch 4/10
570/570 [=====] - 0s 57us/step - loss: 0.3710 - acc: 0.8930 - val_loss: 0.3710
Epoch 5/10
570/570 [=====] - 0s 58us/step - loss: 0.1553 - acc: 0.9737 - val_loss: 0.1553
Epoch 6/10
570/570 [=====] - 0s 54us/step - loss: 0.0640 - acc: 0.9912 - val_loss: 0.0640
Epoch 7/10
570/570 [=====] - 0s 60us/step - loss: 0.0219 - acc: 1.0000 - val_loss: 0.0219
Epoch 8/10
570/570 [=====] - 0s 63us/step - loss: 0.0132 - acc: 1.0000 - val_loss: 0.0132
Epoch 9/10
570/570 [=====] - 0s 60us/step - loss: 0.0071 - acc: 1.0000 - val_loss: 0.0071
Epoch 10/10
570/570 [=====] - 0s 57us/step - loss: 0.0046 - acc: 1.0000 - val_loss: 0.0046

```

```

In [30]: # Determine the npc and nnode that provides the highest validation accuracy
# TO DO
highest_accuracy = result[0][0]
opt_npc_index = 0
opt_nnode_index = 0
for i in range(0,len(npcs)):
    for j in range(0,len(nnodes)):
        if result[i][j] > highest_accuracy:
            highest_accuracy = result[i][j]
            opt_npc_index = i
            opt_nnode_index = j
print("The best npc is %d, and the best nnode is %d." % (npcs[opt_npc_index],nnodes[opt_nnode_index]))
print("The best validation accuracy is %f." % highest_accuracy)

```

The best npc is 50, and the best nnode is 250.
The best validation accuracy is 0.792982.

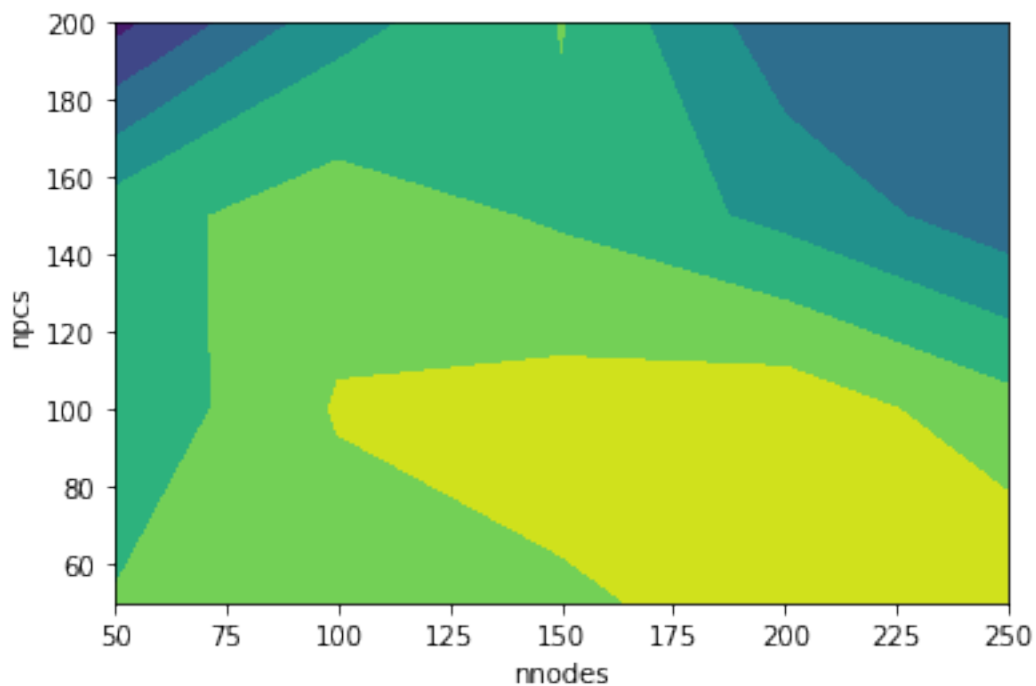
```

In [31]: # Produce a contour plot of the accuracy using different nnode and npc combinations
# TO DO

# plt.contourf ...
grid_x, grid_y = np.mgrid[50:250:50, 50:300:50]
plt.contourf(grid_y,grid_x,result)
plt.xlabel("nnodes")
plt.ylabel("npcs")

```

Out[31]: Text(0,0.5,'npcs')



1.1 Now let us compare the PCA+NN with applying a CNN on the raw image data only.

Note that you should scale your image data to between 0 and 1. And you should reshape your training and testing data according to image width and height

```
In [35]: # Data preparation for input to CNN
# TO DO
Xtr_cnn = X_train.astype("float32")/255
Xts_cnn = X_test.astype("float32")/255
Xtr_cnn = np.reshape(Xtr_cnn, (len(Xtr_cnn),h,w,1))
Xts_cnn = np.reshape(Xts_cnn, (len(X)-len(Xtr_cnn),h,w,1))

In [36]: # Set up a CNN model
# You can use 2 conv2D layer, each with kernel size of 5x5, each followed by a pooling.
# For this part, let both conv2D layer generate 16 channels.
# The Conv layer should be followed by a flatten layer and two dense layers.
# The first dense layer should produce 200 outputs.
# The last dense layer is the output layer with n_classes output using 'softmax' activation.
# Print model summary to verify it follows the desired structure and compile the model.

# TO DO
model = Sequential()
model.add(Conv2D(16, (5, 5),
                padding='valid',
                input_shape=Xtr_cnn.shape[1:],
                activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(16, (5, 5), padding='valid', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(200, activation='relu'))
model.add(Dense(nout, activation='softmax')) # TO DO
print(model.summary())
```

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 46, 33, 16)	416
max_pooling2d_1 (MaxPooling2D)	(None, 23, 16, 16)	0
conv2d_2 (Conv2D)	(None, 19, 12, 16)	6416
max_pooling2d_2 (MaxPooling2D)	(None, 9, 6, 16)	0

flatten_1 (Flatten)	(None, 864)	0

dense_1 (Dense)	(None, 200)	173000

dense_2 (Dense)	(None, 5)	1005
=====		
Total params: 180,837		
Trainable params: 180,837		
Non-trainable params: 0		

None		

```
In [37]: # Fit the model using batch size=100, epochs = 40
        # Print the accuracy on the validation set
```

```
# TO DO
opt = optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=1e-08, decay=0.0)
# Let's train the model using Adam
model.compile(loss='sparse_categorical_crossentropy',
              optimizer=opt, metrics=['accuracy'])
hist_basic = model.fit(Xtr_cnn, y_train, batch_size=100, epochs=40,
                      validation_data=(Xts_cnn, y_test), shuffle=True)
print("The accuracy on validation set is:")
print(hist_basic.history['val_acc'])
```

Train on 570 samples, validate on 570 samples

Epoch 1/40

570/570 [=====] - 2s 3ms/step - loss: 1.4842 - acc: 0.3877 - val_loss

Epoch 2/40

570/570 [=====] - 1s 3ms/step - loss: 1.3941 - acc: 0.4649 - val_loss

Epoch 3/40

570/570 [=====] - 2s 3ms/step - loss: 1.3714 - acc: 0.4649 - val_loss

Epoch 4/40

570/570 [=====] - 1s 3ms/step - loss: 1.3443 - acc: 0.4649 - val_loss

Epoch 5/40

570/570 [=====] - 1s 3ms/step - loss: 1.3016 - acc: 0.4912 - val_loss

Epoch 6/40

570/570 [=====] - 2s 3ms/step - loss: 1.2390 - acc: 0.5000 - val_loss

Epoch 7/40

570/570 [=====] - 2s 3ms/step - loss: 1.1645 - acc: 0.5579 - val_loss

Epoch 8/40

570/570 [=====] - 1s 3ms/step - loss: 1.0714 - acc: 0.6123 - val_loss

Epoch 9/40

570/570 [=====] - 1s 2ms/step - loss: 1.0132 - acc: 0.6404 - val_loss

Epoch 10/40

570/570 [=====] - 1s 3ms/step - loss: 0.9418 - acc: 0.6561 - val_loss

Epoch 11/40

570/570 [=====] - 1s 2ms/step - loss: 0.8758 - acc: 0.6842 - val_loss
Epoch 12/40
570/570 [=====] - 1s 2ms/step - loss: 0.8227 - acc: 0.7158 - val_loss
Epoch 13/40
570/570 [=====] - 1s 2ms/step - loss: 0.7655 - acc: 0.7386 - val_loss
Epoch 14/40
570/570 [=====] - 1s 3ms/step - loss: 0.6990 - acc: 0.7632 - val_loss
Epoch 15/40
570/570 [=====] - 1s 2ms/step - loss: 0.6613 - acc: 0.7860 - val_loss
Epoch 16/40
570/570 [=====] - 1s 2ms/step - loss: 0.6094 - acc: 0.8105 - val_loss
Epoch 17/40
570/570 [=====] - 1s 2ms/step - loss: 0.5817 - acc: 0.8263 - val_loss
Epoch 18/40
570/570 [=====] - 1s 2ms/step - loss: 0.5340 - acc: 0.8456 - val_loss
Epoch 19/40
570/570 [=====] - 2s 3ms/step - loss: 0.4920 - acc: 0.8544 - val_loss
Epoch 20/40
570/570 [=====] - 1s 2ms/step - loss: 0.4560 - acc: 0.8632 - val_loss
Epoch 21/40
570/570 [=====] - 2s 3ms/step - loss: 0.4495 - acc: 0.8579 - val_loss
Epoch 22/40
570/570 [=====] - 1s 3ms/step - loss: 0.4474 - acc: 0.8491 - val_loss
Epoch 23/40
570/570 [=====] - 1s 3ms/step - loss: 0.4532 - acc: 0.8544 - val_loss
Epoch 24/40
570/570 [=====] - 1s 2ms/step - loss: 0.4010 - acc: 0.8807 - val_loss
Epoch 25/40
570/570 [=====] - 1s 2ms/step - loss: 0.3669 - acc: 0.8965 - val_loss
Epoch 26/40
570/570 [=====] - 1s 2ms/step - loss: 0.3661 - acc: 0.8825 - val_loss
Epoch 27/40
570/570 [=====] - 2s 3ms/step - loss: 0.3955 - acc: 0.8719 - val_loss
Epoch 28/40
570/570 [=====] - 1s 3ms/step - loss: 0.3784 - acc: 0.8754 - val_loss
Epoch 29/40
570/570 [=====] - 1s 3ms/step - loss: 0.3359 - acc: 0.8965 - val_loss
Epoch 30/40
570/570 [=====] - 1s 2ms/step - loss: 0.3253 - acc: 0.9105 - val_loss
Epoch 31/40
570/570 [=====] - 1s 2ms/step - loss: 0.3294 - acc: 0.9000 - val_loss
Epoch 32/40
570/570 [=====] - 1s 2ms/step - loss: 0.3716 - acc: 0.8772 - val_loss
Epoch 33/40
570/570 [=====] - 1s 3ms/step - loss: 0.3400 - acc: 0.8912 - val_loss
Epoch 34/40
570/570 [=====] - 1s 2ms/step - loss: 0.3221 - acc: 0.9035 - val_loss
Epoch 35/40

```

570/570 [=====] - 1s 2ms/step - loss: 0.3049 - acc: 0.9018 - val_loss
Epoch 36/40
570/570 [=====] - 1s 3ms/step - loss: 0.2915 - acc: 0.9158 - val_loss
Epoch 37/40
570/570 [=====] - 1s 2ms/step - loss: 0.2561 - acc: 0.9316 - val_loss
Epoch 38/40
570/570 [=====] - 1s 2ms/step - loss: 0.2444 - acc: 0.9368 - val_loss
Epoch 39/40
570/570 [=====] - 1s 2ms/step - loss: 0.2327 - acc: 0.9421 - val_loss
Epoch 40/40
570/570 [=====] - 1s 2ms/step - loss: 0.2227 - acc: 0.9404 - val_loss
The accuracy on validation set is:
[0.46491228070175439, 0.46491228070175439, 0.46491228070175439, 0.46491228070175439, 0.4684210

```

How do the result compared with the PCA+NN method? (If you did right, they should be similar, with PCA+NN being slightly better. If you used more training data (e.g. 75%) and you trained the CNN with more epochs, CNN method may get better).

A: PCA+NN method is slightly better with validation accuracy around 0.83, CNN method is with validation accuracy around 0.79.

1.2 Repeat the above using a small dataset

Instead of using 50% of the total data for training, let us assume you have only 10% of the total data for training. Repeat both the PCA+NN and the CNN method, to see which one gives you better results.

Note that with only 10% data for training, the range of the npc has to be set to be below the total number of training samples.

For the CNN model, because you have small number of training samples, you cannot train a network with a large number of parameters reliably. Instead of producing 16 channels for each of the two conv2D layers, configure the model to produce only 8 channels each.

In [41]: *## TO DO*

```

X_train, X_test, y_train, y_test = train_test_split(
    X, y, stratify=y, test_size=0.5)
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify = y, test_size=0.9)
n_samples, _ = X_train.shape
Xtr_mean = np.mean(X_train, 0)
Xtr = X_train - Xtr_mean[None, :]
Utr, Str, Vtr = np.linalg.svd(Xtr, full_matrices=False)
nnodes = [50, 100, 150, 200, 250]
npcs = [50, 60, 70, 80, 90, 100]
result = np.zeros((len(npcs), len(nnodes)))
loss_hist = []
train_acc_hist = []
val_acc_hist = []
for i, npc in enumerate(npcs):
    for j, nnode in enumerate(nnodes):

```

```

K.clear_session()
eigenface = Vtr[:,npc,:]
Xtr_pca = Xtr.dot(eigenface.T)
Xtr_pca_s = Xtr_pca / Str[None,:npc] * np.sqrt(n_samples)
Xts = X_test - Xtr_mean[None,:]
Xts_pca = Xts.dot(eigenface.T)
Xts_pca_s = Xts_pca / Str[None,:npc] * np.sqrt(n_samples)
nin = Xtr_pca.shape[1] # dimension of input data
nh = nnode # number of hidden units
nout = int(np.max(y_train)+1)
model = Sequential()
model.add(Dense(nh, input_shape=(nin,), activation='relu', name='hidden'))
model.add(Dense(nout, activation='softmax', name='output'))
opt = optimizers.Adam(lr=0.01, beta_1=0.9, beta_2=0.999, epsilon=1e-08, decay=0)
model.compile(optimizer=opt,loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
hist = model.fit(Xtr_pca_s, y_train, epochs=10, batch_size=100,
                 validation_data=(Xts_pca_s, y_test))
result[i][j] = hist.history['val_acc'][-1]
highest_accuracy = result[0][0]
opt_npc_index = 0
opt_nnode_index = 0
for i in range(0,len(npcs)):
    for j in range(0,len(nnodes)):
        if result[i][j] > highest_accuracy:
            highest_accuracy = result[i][j]
            opt_npc_index = i
            opt_nnode_index = j
print("The best npc is %d, and the best nnode is %d." % (npcs[opt_npc_index],nnodes[opt_nnode_index]))
print("The best validation accuracy is %f." % highest_accuracy)

```

Train on 114 samples, validate on 1026 samples

Epoch 1/10

114/114 [=====] - 0s 2ms/step - loss: 1.6904 - acc: 0.2982 - val_loss: 1.6904

Epoch 2/10

114/114 [=====] - 0s 198us/step - loss: 1.2262 - acc: 0.5088 - val_loss: 1.2262

Epoch 3/10

114/114 [=====] - 0s 205us/step - loss: 0.9618 - acc: 0.7018 - val_loss: 0.9618

Epoch 4/10

114/114 [=====] - 0s 151us/step - loss: 0.7583 - acc: 0.7982 - val_loss: 0.7583

Epoch 5/10

114/114 [=====] - 0s 205us/step - loss: 0.6010 - acc: 0.8684 - val_loss: 0.6010

Epoch 6/10

114/114 [=====] - 0s 179us/step - loss: 0.4785 - acc: 0.9211 - val_loss: 0.4785

Epoch 7/10

114/114 [=====] - 0s 158us/step - loss: 0.3843 - acc: 0.9298 - val_loss: 0.3843

Epoch 8/10

114/114 [=====] - 0s 180us/step - loss: 0.3067 - acc: 0.9386 - val_loss: 0.3067

```

Epoch 9/10
114/114 [=====] - 0s 173us/step - loss: 0.2408 - acc: 0.9737 - val_loss: 0.2408
Epoch 10/10
114/114 [=====] - 0s 156us/step - loss: 0.1897 - acc: 0.9737 - val_loss: 0.1897
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.6233 - acc: 0.2719 - val_loss: 1.6233
Epoch 2/10
114/114 [=====] - 0s 158us/step - loss: 0.9601 - acc: 0.7018 - val_loss: 0.9601
Epoch 3/10
114/114 [=====] - 0s 177us/step - loss: 0.6531 - acc: 0.8070 - val_loss: 0.6531
Epoch 4/10
114/114 [=====] - 0s 195us/step - loss: 0.4570 - acc: 0.9035 - val_loss: 0.4570
Epoch 5/10
114/114 [=====] - 0s 156us/step - loss: 0.3270 - acc: 0.9561 - val_loss: 0.3270
Epoch 6/10
114/114 [=====] - 0s 163us/step - loss: 0.2350 - acc: 0.9649 - val_loss: 0.2350
Epoch 7/10
114/114 [=====] - 0s 181us/step - loss: 0.1646 - acc: 0.9912 - val_loss: 0.1646
Epoch 8/10
114/114 [=====] - 0s 154us/step - loss: 0.1132 - acc: 1.0000 - val_loss: 0.1132
Epoch 9/10
114/114 [=====] - 0s 185us/step - loss: 0.0778 - acc: 1.0000 - val_loss: 0.0778
Epoch 10/10
114/114 [=====] - 0s 182us/step - loss: 0.0537 - acc: 1.0000 - val_loss: 0.0537
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.7323 - acc: 0.2368 - val_loss: 1.7323
Epoch 2/10
114/114 [=====] - 0s 161us/step - loss: 0.9330 - acc: 0.6579 - val_loss: 0.9330
Epoch 3/10
114/114 [=====] - 0s 205us/step - loss: 0.5822 - acc: 0.8509 - val_loss: 0.5822
Epoch 4/10
114/114 [=====] - 0s 191us/step - loss: 0.3773 - acc: 0.9474 - val_loss: 0.3773
Epoch 5/10
114/114 [=====] - 0s 190us/step - loss: 0.2443 - acc: 0.9912 - val_loss: 0.2443
Epoch 6/10
114/114 [=====] - 0s 387us/step - loss: 0.1591 - acc: 0.9912 - val_loss: 0.1591
Epoch 7/10
114/114 [=====] - 0s 321us/step - loss: 0.0999 - acc: 1.0000 - val_loss: 0.0999
Epoch 8/10
114/114 [=====] - 0s 271us/step - loss: 0.0641 - acc: 1.0000 - val_loss: 0.0641
Epoch 9/10
114/114 [=====] - 0s 251us/step - loss: 0.0425 - acc: 1.0000 - val_loss: 0.0425
Epoch 10/10
114/114 [=====] - 0s 214us/step - loss: 0.0293 - acc: 1.0000 - val_loss: 0.0293
Train on 114 samples, validate on 1026 samples
Epoch 1/10

```

```

114/114 [=====] - 0s 2ms/step - loss: 1.6380 - acc: 0.2632 - val_loss
Epoch 2/10
114/114 [=====] - 0s 161us/step - loss: 0.7516 - acc: 0.7544 - val_loss
Epoch 3/10
114/114 [=====] - 0s 173us/step - loss: 0.4446 - acc: 0.8860 - val_loss
Epoch 4/10
114/114 [=====] - 0s 158us/step - loss: 0.2516 - acc: 0.9561 - val_loss
Epoch 5/10
114/114 [=====] - 0s 186us/step - loss: 0.1564 - acc: 0.9825 - val_loss
Epoch 6/10
114/114 [=====] - 0s 170us/step - loss: 0.1081 - acc: 0.9825 - val_loss
Epoch 7/10
114/114 [=====] - 0s 208us/step - loss: 0.0624 - acc: 0.9825 - val_loss
Epoch 8/10
114/114 [=====] - 0s 193us/step - loss: 0.0377 - acc: 0.9912 - val_loss
Epoch 9/10
114/114 [=====] - 0s 294us/step - loss: 0.0252 - acc: 1.0000 - val_loss
Epoch 10/10
114/114 [=====] - 0s 306us/step - loss: 0.0182 - acc: 1.0000 - val_loss
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.6684 - acc: 0.2018 - val_loss
Epoch 2/10
114/114 [=====] - 0s 179us/step - loss: 0.7468 - acc: 0.7281 - val_loss
Epoch 3/10
114/114 [=====] - 0s 222us/step - loss: 0.4239 - acc: 0.8947 - val_loss
Epoch 4/10
114/114 [=====] - 0s 186us/step - loss: 0.2361 - acc: 0.9561 - val_loss
Epoch 5/10
114/114 [=====] - 0s 189us/step - loss: 0.1378 - acc: 0.9737 - val_loss
Epoch 6/10
114/114 [=====] - 0s 162us/step - loss: 0.0802 - acc: 1.0000 - val_loss
Epoch 7/10
114/114 [=====] - 0s 186us/step - loss: 0.0545 - acc: 1.0000 - val_loss
Epoch 8/10
114/114 [=====] - 0s 216us/step - loss: 0.0354 - acc: 1.0000 - val_loss
Epoch 9/10
114/114 [=====] - 0s 200us/step - loss: 0.0240 - acc: 1.0000 - val_loss
Epoch 10/10
114/114 [=====] - 0s 202us/step - loss: 0.0161 - acc: 1.0000 - val_loss
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 2.0253 - acc: 0.2456 - val_loss
Epoch 2/10
114/114 [=====] - 0s 195us/step - loss: 1.2639 - acc: 0.4737 - val_loss
Epoch 3/10
114/114 [=====] - 0s 249us/step - loss: 0.8926 - acc: 0.6930 - val_loss
Epoch 4/10

```

```

114/114 [=====] - 0s 183us/step - loss: 0.6819 - acc: 0.7895 - val_loss: 0.6819
Epoch 5/10
114/114 [=====] - 0s 235us/step - loss: 0.5304 - acc: 0.8596 - val_loss: 0.5304
Epoch 6/10
114/114 [=====] - 0s 198us/step - loss: 0.4093 - acc: 0.8947 - val_loss: 0.4093
Epoch 7/10
114/114 [=====] - 0s 248us/step - loss: 0.3062 - acc: 0.9211 - val_loss: 0.3062
Epoch 8/10
114/114 [=====] - 0s 274us/step - loss: 0.2269 - acc: 0.9649 - val_loss: 0.2269
Epoch 9/10
114/114 [=====] - 0s 208us/step - loss: 0.1705 - acc: 0.9912 - val_loss: 0.1705
Epoch 10/10
114/114 [=====] - 0s 157us/step - loss: 0.1264 - acc: 1.0000 - val_loss: 0.1264
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.9153 - acc: 0.1930 - val_loss: 1.9153
Epoch 2/10
114/114 [=====] - 0s 159us/step - loss: 1.0128 - acc: 0.6491 - val_loss: 1.0128
Epoch 3/10
114/114 [=====] - 0s 184us/step - loss: 0.6759 - acc: 0.7632 - val_loss: 0.6759
Epoch 4/10
114/114 [=====] - 0s 188us/step - loss: 0.4772 - acc: 0.8333 - val_loss: 0.4772
Epoch 5/10
114/114 [=====] - 0s 205us/step - loss: 0.3117 - acc: 0.9561 - val_loss: 0.3117
Epoch 6/10
114/114 [=====] - 0s 188us/step - loss: 0.1990 - acc: 0.9825 - val_loss: 0.1990
Epoch 7/10
114/114 [=====] - 0s 207us/step - loss: 0.1256 - acc: 0.9912 - val_loss: 0.1256
Epoch 8/10
114/114 [=====] - 0s 159us/step - loss: 0.0857 - acc: 0.9912 - val_loss: 0.0857
Epoch 9/10
114/114 [=====] - 0s 160us/step - loss: 0.0591 - acc: 1.0000 - val_loss: 0.0591
Epoch 10/10
114/114 [=====] - 0s 210us/step - loss: 0.0422 - acc: 1.0000 - val_loss: 0.0422
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 2.1994 - acc: 0.1053 - val_loss: 2.1994
Epoch 2/10
114/114 [=====] - 0s 160us/step - loss: 0.9814 - acc: 0.7807 - val_loss: 0.9814
Epoch 3/10
114/114 [=====] - 0s 210us/step - loss: 0.6070 - acc: 0.7895 - val_loss: 0.6070
Epoch 4/10
114/114 [=====] - 0s 246us/step - loss: 0.4192 - acc: 0.8333 - val_loss: 0.4192
Epoch 5/10
114/114 [=====] - 0s 224us/step - loss: 0.2615 - acc: 0.9386 - val_loss: 0.2615
Epoch 6/10
114/114 [=====] - 0s 384us/step - loss: 0.1548 - acc: 0.9737 - val_loss: 0.1548
Epoch 7/10

```

```

114/114 [=====] - 0s 269us/step - loss: 0.0985 - acc: 0.9912 - val_loss: 0.0985
Epoch 8/10
114/114 [=====] - 0s 278us/step - loss: 0.0662 - acc: 0.9912 - val_loss: 0.0662
Epoch 9/10
114/114 [=====] - 0s 271us/step - loss: 0.0482 - acc: 1.0000 - val_loss: 0.0482
Epoch 10/10
114/114 [=====] - 0s 312us/step - loss: 0.0354 - acc: 1.0000 - val_loss: 0.0354
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.6396 - acc: 0.3070 - val_loss: 1.6396
Epoch 2/10
114/114 [=====] - 0s 187us/step - loss: 0.7322 - acc: 0.7281 - val_loss: 0.7322
Epoch 3/10
114/114 [=====] - 0s 199us/step - loss: 0.4317 - acc: 0.8772 - val_loss: 0.4317
Epoch 4/10
114/114 [=====] - 0s 197us/step - loss: 0.2352 - acc: 0.9737 - val_loss: 0.2352
Epoch 5/10
114/114 [=====] - 0s 210us/step - loss: 0.1195 - acc: 0.9912 - val_loss: 0.1195
Epoch 6/10
114/114 [=====] - 0s 197us/step - loss: 0.0631 - acc: 1.0000 - val_loss: 0.0631
Epoch 7/10
114/114 [=====] - 0s 191us/step - loss: 0.0370 - acc: 1.0000 - val_loss: 0.0370
Epoch 8/10
114/114 [=====] - 0s 204us/step - loss: 0.0235 - acc: 1.0000 - val_loss: 0.0235
Epoch 9/10
114/114 [=====] - 0s 228us/step - loss: 0.0162 - acc: 1.0000 - val_loss: 0.0162
Epoch 10/10
114/114 [=====] - 0s 231us/step - loss: 0.0110 - acc: 1.0000 - val_loss: 0.0110
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.6465 - acc: 0.2982 - val_loss: 1.6465
Epoch 2/10
114/114 [=====] - 0s 268us/step - loss: 0.6836 - acc: 0.6930 - val_loss: 0.6836
Epoch 3/10
114/114 [=====] - 0s 245us/step - loss: 0.3843 - acc: 0.9035 - val_loss: 0.3843
Epoch 4/10
114/114 [=====] - 0s 297us/step - loss: 0.2022 - acc: 0.9737 - val_loss: 0.2022
Epoch 5/10
114/114 [=====] - 0s 253us/step - loss: 0.1072 - acc: 0.9912 - val_loss: 0.1072
Epoch 6/10
114/114 [=====] - 0s 218us/step - loss: 0.0609 - acc: 0.9912 - val_loss: 0.0609
Epoch 7/10
114/114 [=====] - 0s 210us/step - loss: 0.0347 - acc: 1.0000 - val_loss: 0.0347
Epoch 8/10
114/114 [=====] - 0s 254us/step - loss: 0.0212 - acc: 1.0000 - val_loss: 0.0212
Epoch 9/10
114/114 [=====] - 0s 260us/step - loss: 0.0135 - acc: 1.0000 - val_loss: 0.0135
Epoch 10/10

```



```

114/114 [=====] - 0s 175us/step - loss: 0.0097 - acc: 1.0000 - val_loss: 0.0097
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.7950 - acc: 0.2544 - val_loss: 1.7950
Epoch 2/10
114/114 [=====] - 0s 200us/step - loss: 1.1355 - acc: 0.5614 - val_loss: 1.1355
Epoch 3/10
114/114 [=====] - 0s 166us/step - loss: 0.8220 - acc: 0.7368 - val_loss: 0.8220
Epoch 4/10
114/114 [=====] - 0s 152us/step - loss: 0.6162 - acc: 0.8158 - val_loss: 0.6162
Epoch 5/10
114/114 [=====] - 0s 140us/step - loss: 0.4637 - acc: 0.8860 - val_loss: 0.4637
Epoch 6/10
114/114 [=====] - 0s 165us/step - loss: 0.3420 - acc: 0.9298 - val_loss: 0.3420
Epoch 7/10
114/114 [=====] - 0s 182us/step - loss: 0.2490 - acc: 0.9825 - val_loss: 0.2490
Epoch 8/10
114/114 [=====] - 0s 174us/step - loss: 0.1837 - acc: 0.9912 - val_loss: 0.1837
Epoch 9/10
114/114 [=====] - 0s 151us/step - loss: 0.1387 - acc: 1.0000 - val_loss: 0.1387
Epoch 10/10
114/114 [=====] - 0s 224us/step - loss: 0.1055 - acc: 1.0000 - val_loss: 0.1055
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 4ms/step - loss: 1.6452 - acc: 0.2719 - val_loss: 1.6452
Epoch 2/10
114/114 [=====] - 0s 180us/step - loss: 0.8722 - acc: 0.6667 - val_loss: 0.8722
Epoch 3/10
114/114 [=====] - 0s 244us/step - loss: 0.5582 - acc: 0.8158 - val_loss: 0.5582
Epoch 4/10
114/114 [=====] - 0s 200us/step - loss: 0.3641 - acc: 0.8772 - val_loss: 0.3641
Epoch 5/10
114/114 [=====] - 0s 191us/step - loss: 0.2249 - acc: 0.9825 - val_loss: 0.2249
Epoch 6/10
114/114 [=====] - 0s 196us/step - loss: 0.1385 - acc: 1.0000 - val_loss: 0.1385
Epoch 7/10
114/114 [=====] - 0s 169us/step - loss: 0.0810 - acc: 1.0000 - val_loss: 0.0810
Epoch 8/10
114/114 [=====] - 0s 196us/step - loss: 0.0498 - acc: 1.0000 - val_loss: 0.0498
Epoch 9/10
114/114 [=====] - 0s 143us/step - loss: 0.0324 - acc: 1.0000 - val_loss: 0.0324
Epoch 10/10
114/114 [=====] - 0s 185us/step - loss: 0.0220 - acc: 1.0000 - val_loss: 0.0220
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.8350 - acc: 0.1930 - val_loss: 1.8350
Epoch 2/10
114/114 [=====] - 0s 170us/step - loss: 0.8478 - acc: 0.7105 - val_loss: 0.8478

```

```

Epoch 3/10
114/114 [=====] - 0s 179us/step - loss: 0.5193 - acc: 0.8509 - val_loss: 0.5193
Epoch 4/10
114/114 [=====] - 0s 183us/step - loss: 0.3003 - acc: 0.9298 - val_loss: 0.3003
Epoch 5/10
114/114 [=====] - 0s 182us/step - loss: 0.1767 - acc: 0.9825 - val_loss: 0.1767
Epoch 6/10
114/114 [=====] - 0s 220us/step - loss: 0.1097 - acc: 0.9912 - val_loss: 0.1097
Epoch 7/10
114/114 [=====] - 0s 205us/step - loss: 0.0717 - acc: 0.9912 - val_loss: 0.0717
Epoch 8/10
114/114 [=====] - 0s 203us/step - loss: 0.0486 - acc: 1.0000 - val_loss: 0.0486
Epoch 9/10
114/114 [=====] - 0s 247us/step - loss: 0.0339 - acc: 1.0000 - val_loss: 0.0339
Epoch 10/10
114/114 [=====] - 0s 262us/step - loss: 0.0245 - acc: 1.0000 - val_loss: 0.0245
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.8324 - acc: 0.1667 - val_loss: 1.8324
Epoch 2/10
114/114 [=====] - 0s 231us/step - loss: 0.7643 - acc: 0.6842 - val_loss: 0.7643
Epoch 3/10
114/114 [=====] - 0s 209us/step - loss: 0.4110 - acc: 0.8772 - val_loss: 0.4110
Epoch 4/10
114/114 [=====] - 0s 239us/step - loss: 0.2029 - acc: 0.9825 - val_loss: 0.2029
Epoch 5/10
114/114 [=====] - 0s 236us/step - loss: 0.1121 - acc: 1.0000 - val_loss: 0.1121
Epoch 6/10
114/114 [=====] - 0s 349us/step - loss: 0.0677 - acc: 1.0000 - val_loss: 0.0677
Epoch 7/10
114/114 [=====] - 0s 261us/step - loss: 0.0426 - acc: 1.0000 - val_loss: 0.0426
Epoch 8/10
114/114 [=====] - 0s 286us/step - loss: 0.0270 - acc: 1.0000 - val_loss: 0.0270
Epoch 9/10
114/114 [=====] - 0s 285us/step - loss: 0.0174 - acc: 1.0000 - val_loss: 0.0174
Epoch 10/10
114/114 [=====] - 0s 214us/step - loss: 0.0112 - acc: 1.0000 - val_loss: 0.0112
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.7138 - acc: 0.2982 - val_loss: 1.7138
Epoch 2/10
114/114 [=====] - 0s 167us/step - loss: 0.6811 - acc: 0.7105 - val_loss: 0.6811
Epoch 3/10
114/114 [=====] - 0s 194us/step - loss: 0.3587 - acc: 0.8947 - val_loss: 0.3587
Epoch 4/10
114/114 [=====] - 0s 219us/step - loss: 0.1739 - acc: 0.9825 - val_loss: 0.1739
Epoch 5/10
114/114 [=====] - 0s 219us/step - loss: 0.0853 - acc: 1.0000 - val_loss: 0.0853

```

```

Epoch 6/10
114/114 [=====] - 0s 239us/step - loss: 0.0494 - acc: 1.0000 - val_loss: 0.0494
Epoch 7/10
114/114 [=====] - 0s 193us/step - loss: 0.0303 - acc: 1.0000 - val_loss: 0.0303
Epoch 8/10
114/114 [=====] - 0s 243us/step - loss: 0.0186 - acc: 1.0000 - val_loss: 0.0186
Epoch 9/10
114/114 [=====] - 0s 194us/step - loss: 0.0113 - acc: 1.0000 - val_loss: 0.0113
Epoch 10/10
114/114 [=====] - 0s 254us/step - loss: 0.0071 - acc: 1.0000 - val_loss: 0.0071
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 3ms/step - loss: 1.8343 - acc: 0.1842 - val_loss: 1.8343
Epoch 2/10
114/114 [=====] - 0s 181us/step - loss: 1.1308 - acc: 0.6053 - val_loss: 1.1308
Epoch 3/10
114/114 [=====] - 0s 169us/step - loss: 0.8028 - acc: 0.7632 - val_loss: 0.8028
Epoch 4/10
114/114 [=====] - 0s 170us/step - loss: 0.5921 - acc: 0.8596 - val_loss: 0.5921
Epoch 5/10
114/114 [=====] - 0s 186us/step - loss: 0.4357 - acc: 0.9211 - val_loss: 0.4357
Epoch 6/10
114/114 [=====] - 0s 143us/step - loss: 0.3247 - acc: 0.9737 - val_loss: 0.3247
Epoch 7/10
114/114 [=====] - 0s 186us/step - loss: 0.2376 - acc: 0.9825 - val_loss: 0.2376
Epoch 8/10
114/114 [=====] - 0s 165us/step - loss: 0.1746 - acc: 1.0000 - val_loss: 0.1746
Epoch 9/10
114/114 [=====] - 0s 194us/step - loss: 0.1266 - acc: 1.0000 - val_loss: 0.1266
Epoch 10/10
114/114 [=====] - 0s 190us/step - loss: 0.0930 - acc: 1.0000 - val_loss: 0.0930
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 2.5671 - acc: 0.1140 - val_loss: 2.5671
Epoch 2/10
114/114 [=====] - 0s 250us/step - loss: 1.1793 - acc: 0.5702 - val_loss: 1.1793
Epoch 3/10
114/114 [=====] - 0s 206us/step - loss: 0.6385 - acc: 0.8333 - val_loss: 0.6385
Epoch 4/10
114/114 [=====] - 0s 205us/step - loss: 0.4226 - acc: 0.9123 - val_loss: 0.4226
Epoch 5/10
114/114 [=====] - 0s 237us/step - loss: 0.3066 - acc: 0.9298 - val_loss: 0.3066
Epoch 6/10
114/114 [=====] - 0s 264us/step - loss: 0.2083 - acc: 0.9561 - val_loss: 0.2083
Epoch 7/10
114/114 [=====] - 0s 258us/step - loss: 0.1336 - acc: 0.9912 - val_loss: 0.1336
Epoch 8/10
114/114 [=====] - 0s 217us/step - loss: 0.0833 - acc: 1.0000 - val_loss: 0.0833

```

Epoch 9/10
114/114 [=====] - 0s 268us/step - loss: 0.0501 - acc: 1.0000 - val_loss: 0.0501
Epoch 10/10
114/114 [=====] - 0s 207us/step - loss: 0.0330 - acc: 1.0000 - val_loss: 0.0330
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.6092 - acc: 0.4298 - val_loss: 1.6092
Epoch 2/10
114/114 [=====] - 0s 224us/step - loss: 0.6742 - acc: 0.8246 - val_loss: 0.6742
Epoch 3/10
114/114 [=====] - 0s 181us/step - loss: 0.3472 - acc: 0.9737 - val_loss: 0.3472
Epoch 4/10
114/114 [=====] - 0s 172us/step - loss: 0.1884 - acc: 1.0000 - val_loss: 0.1884
Epoch 5/10
114/114 [=====] - 0s 197us/step - loss: 0.1000 - acc: 1.0000 - val_loss: 0.1000
Epoch 6/10
114/114 [=====] - 0s 199us/step - loss: 0.0549 - acc: 1.0000 - val_loss: 0.0549
Epoch 7/10
114/114 [=====] - 0s 208us/step - loss: 0.0310 - acc: 1.0000 - val_loss: 0.0310
Epoch 8/10
114/114 [=====] - 0s 179us/step - loss: 0.0181 - acc: 1.0000 - val_loss: 0.0181
Epoch 9/10
114/114 [=====] - 0s 185us/step - loss: 0.0111 - acc: 1.0000 - val_loss: 0.0111
Epoch 10/10
114/114 [=====] - 0s 183us/step - loss: 0.0071 - acc: 1.0000 - val_loss: 0.0071
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 3ms/step - loss: 2.0338 - acc: 0.1491 - val_loss: 2.0338
Epoch 2/10
114/114 [=====] - 0s 187us/step - loss: 0.7785 - acc: 0.7368 - val_loss: 0.7785
Epoch 3/10
114/114 [=====] - 0s 185us/step - loss: 0.4785 - acc: 0.7895 - val_loss: 0.4785
Epoch 4/10
114/114 [=====] - 0s 210us/step - loss: 0.2627 - acc: 0.9474 - val_loss: 0.2627
Epoch 5/10
114/114 [=====] - 0s 177us/step - loss: 0.1296 - acc: 1.0000 - val_loss: 0.1296
Epoch 6/10
114/114 [=====] - 0s 201us/step - loss: 0.0643 - acc: 1.0000 - val_loss: 0.0643
Epoch 7/10
114/114 [=====] - 0s 185us/step - loss: 0.0348 - acc: 1.0000 - val_loss: 0.0348
Epoch 8/10
114/114 [=====] - 0s 225us/step - loss: 0.0207 - acc: 1.0000 - val_loss: 0.0207
Epoch 9/10
114/114 [=====] - 0s 211us/step - loss: 0.0134 - acc: 1.0000 - val_loss: 0.0134
Epoch 10/10
114/114 [=====] - 0s 211us/step - loss: 0.0093 - acc: 1.0000 - val_loss: 0.0093
Train on 114 samples, validate on 1026 samples
Epoch 1/10

```

114/114 [=====] - 0s 2ms/step - loss: 1.7110 - acc: 0.2281 - val_loss
Epoch 2/10
114/114 [=====] - 0s 176us/step - loss: 0.6111 - acc: 0.7456 - val_loss
Epoch 3/10
114/114 [=====] - 0s 167us/step - loss: 0.2616 - acc: 0.9825 - val_loss
Epoch 4/10
114/114 [=====] - 0s 176us/step - loss: 0.1098 - acc: 1.0000 - val_loss
Epoch 5/10
114/114 [=====] - 0s 200us/step - loss: 0.0555 - acc: 1.0000 - val_loss
Epoch 6/10
114/114 [=====] - 0s 188us/step - loss: 0.0305 - acc: 1.0000 - val_loss
Epoch 7/10
114/114 [=====] - 0s 210us/step - loss: 0.0180 - acc: 1.0000 - val_loss
Epoch 8/10
114/114 [=====] - 0s 240us/step - loss: 0.0109 - acc: 1.0000 - val_loss
Epoch 9/10
114/114 [=====] - 0s 227us/step - loss: 0.0067 - acc: 1.0000 - val_loss
Epoch 10/10
114/114 [=====] - 0s 229us/step - loss: 0.0044 - acc: 1.0000 - val_loss
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 3ms/step - loss: 2.5833 - acc: 0.1316 - val_loss
Epoch 2/10
114/114 [=====] - 0s 239us/step - loss: 1.6094 - acc: 0.3509 - val_loss
Epoch 3/10
114/114 [=====] - 0s 250us/step - loss: 1.0194 - acc: 0.6842 - val_loss
Epoch 4/10
114/114 [=====] - 0s 246us/step - loss: 0.6793 - acc: 0.8684 - val_loss
Epoch 5/10
114/114 [=====] - 0s 332us/step - loss: 0.4856 - acc: 0.9211 - val_loss
Epoch 6/10
114/114 [=====] - 0s 277us/step - loss: 0.3735 - acc: 0.9298 - val_loss
Epoch 7/10
114/114 [=====] - 0s 216us/step - loss: 0.2945 - acc: 0.9298 - val_loss
Epoch 8/10
114/114 [=====] - 0s 254us/step - loss: 0.2224 - acc: 0.9561 - val_loss
Epoch 9/10
114/114 [=====] - 0s 247us/step - loss: 0.1610 - acc: 0.9825 - val_loss
Epoch 10/10
114/114 [=====] - 0s 223us/step - loss: 0.1120 - acc: 0.9912 - val_loss
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 2.1399 - acc: 0.2105 - val_loss
Epoch 2/10
114/114 [=====] - 0s 187us/step - loss: 0.9121 - acc: 0.7018 - val_loss
Epoch 3/10
114/114 [=====] - 0s 255us/step - loss: 0.5221 - acc: 0.8246 - val_loss
Epoch 4/10

```

```

114/114 [=====] - 0s 279us/step - loss: 0.3185 - acc: 0.9386 - val_loss: 0.3185
Epoch 5/10
114/114 [=====] - 0s 243us/step - loss: 0.1886 - acc: 0.9912 - val_loss: 0.1886
Epoch 6/10
114/114 [=====] - 0s 229us/step - loss: 0.1138 - acc: 1.0000 - val_loss: 0.1138
Epoch 7/10
114/114 [=====] - 0s 241us/step - loss: 0.0701 - acc: 1.0000 - val_loss: 0.0701
Epoch 8/10
114/114 [=====] - 0s 248us/step - loss: 0.0452 - acc: 1.0000 - val_loss: 0.0452
Epoch 9/10
114/114 [=====] - 0s 165us/step - loss: 0.0299 - acc: 1.0000 - val_loss: 0.0299
Epoch 10/10
114/114 [=====] - 0s 271us/step - loss: 0.0214 - acc: 1.0000 - val_loss: 0.0214
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.8167 - acc: 0.1842 - val_loss: 1.8167
Epoch 2/10
114/114 [=====] - 0s 170us/step - loss: 0.7433 - acc: 0.7982 - val_loss: 0.7433
Epoch 3/10
114/114 [=====] - 0s 181us/step - loss: 0.3647 - acc: 0.9474 - val_loss: 0.3647
Epoch 4/10
114/114 [=====] - 0s 178us/step - loss: 0.1727 - acc: 0.9912 - val_loss: 0.1727
Epoch 5/10
114/114 [=====] - 0s 173us/step - loss: 0.0875 - acc: 1.0000 - val_loss: 0.0875
Epoch 6/10
114/114 [=====] - 0s 182us/step - loss: 0.0484 - acc: 1.0000 - val_loss: 0.0484
Epoch 7/10
114/114 [=====] - 0s 177us/step - loss: 0.0295 - acc: 1.0000 - val_loss: 0.0295
Epoch 8/10
114/114 [=====] - 0s 178us/step - loss: 0.0185 - acc: 1.0000 - val_loss: 0.0185
Epoch 9/10
114/114 [=====] - 0s 192us/step - loss: 0.0123 - acc: 1.0000 - val_loss: 0.0123
Epoch 10/10
114/114 [=====] - 0s 197us/step - loss: 0.0082 - acc: 1.0000 - val_loss: 0.0082
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 3ms/step - loss: 1.6185 - acc: 0.2895 - val_loss: 1.6185
Epoch 2/10
114/114 [=====] - 0s 246us/step - loss: 0.5094 - acc: 0.9211 - val_loss: 0.5094
Epoch 3/10
114/114 [=====] - 0s 273us/step - loss: 0.2204 - acc: 0.9649 - val_loss: 0.2204
Epoch 4/10
114/114 [=====] - 0s 289us/step - loss: 0.1051 - acc: 1.0000 - val_loss: 0.1051
Epoch 5/10
114/114 [=====] - 0s 355us/step - loss: 0.0529 - acc: 1.0000 - val_loss: 0.0529
Epoch 6/10
114/114 [=====] - 0s 317us/step - loss: 0.0289 - acc: 1.0000 - val_loss: 0.0289
Epoch 7/10

```

```

114/114 [=====] - 0s 253us/step - loss: 0.0174 - acc: 1.0000 - val_loss: 0.0174
Epoch 8/10
114/114 [=====] - 0s 208us/step - loss: 0.0105 - acc: 1.0000 - val_loss: 0.0105
Epoch 9/10
114/114 [=====] - 0s 263us/step - loss: 0.0070 - acc: 1.0000 - val_loss: 0.0070
Epoch 10/10
114/114 [=====] - 0s 230us/step - loss: 0.0046 - acc: 1.0000 - val_loss: 0.0046
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.8944 - acc: 0.1404 - val_loss: 1.8944
Epoch 2/10
114/114 [=====] - 0s 197us/step - loss: 0.5178 - acc: 0.8947 - val_loss: 0.5178
Epoch 3/10
114/114 [=====] - 0s 257us/step - loss: 0.2081 - acc: 0.9912 - val_loss: 0.2081
Epoch 4/10
114/114 [=====] - 0s 273us/step - loss: 0.0828 - acc: 1.0000 - val_loss: 0.0828
Epoch 5/10
114/114 [=====] - 0s 252us/step - loss: 0.0370 - acc: 1.0000 - val_loss: 0.0370
Epoch 6/10
114/114 [=====] - 0s 218us/step - loss: 0.0199 - acc: 1.0000 - val_loss: 0.0199
Epoch 7/10
114/114 [=====] - 0s 190us/step - loss: 0.0118 - acc: 1.0000 - val_loss: 0.0118
Epoch 8/10
114/114 [=====] - 0s 300us/step - loss: 0.0077 - acc: 1.0000 - val_loss: 0.0077
Epoch 9/10
114/114 [=====] - 0s 244us/step - loss: 0.0053 - acc: 1.0000 - val_loss: 0.0053
Epoch 10/10
114/114 [=====] - 0s 232us/step - loss: 0.0038 - acc: 1.0000 - val_loss: 0.0038
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 3ms/step - loss: 2.0177 - acc: 0.1930 - val_loss: 2.0177
Epoch 2/10
114/114 [=====] - 0s 170us/step - loss: 1.1667 - acc: 0.5088 - val_loss: 1.1667
Epoch 3/10
114/114 [=====] - 0s 203us/step - loss: 0.7713 - acc: 0.7807 - val_loss: 0.7713
Epoch 4/10
114/114 [=====] - 0s 225us/step - loss: 0.5577 - acc: 0.8772 - val_loss: 0.5577
Epoch 5/10
114/114 [=====] - 0s 224us/step - loss: 0.4066 - acc: 0.9123 - val_loss: 0.4066
Epoch 6/10
114/114 [=====] - 0s 226us/step - loss: 0.2954 - acc: 0.9474 - val_loss: 0.2954
Epoch 7/10
114/114 [=====] - 0s 201us/step - loss: 0.2181 - acc: 0.9649 - val_loss: 0.2181
Epoch 8/10
114/114 [=====] - 0s 205us/step - loss: 0.1617 - acc: 0.9737 - val_loss: 0.1617
Epoch 9/10
114/114 [=====] - 0s 202us/step - loss: 0.1192 - acc: 0.9912 - val_loss: 0.1192
Epoch 10/10

```

```

114/114 [=====] - 0s 217us/step - loss: 0.0872 - acc: 1.0000 - val_loss: 0.0872
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 3ms/step - loss: 2.0224 - acc: 0.1316 - val_loss: 0.0872
Epoch 2/10
114/114 [=====] - 0s 147us/step - loss: 0.8256 - acc: 0.7456 - val_loss: 0.0872
Epoch 3/10
114/114 [=====] - 0s 182us/step - loss: 0.4746 - acc: 0.8684 - val_loss: 0.0872
Epoch 4/10
114/114 [=====] - 0s 189us/step - loss: 0.2973 - acc: 0.9211 - val_loss: 0.0872
Epoch 5/10
114/114 [=====] - 0s 210us/step - loss: 0.1755 - acc: 0.9737 - val_loss: 0.0872
Epoch 6/10
114/114 [=====] - 0s 215us/step - loss: 0.1007 - acc: 1.0000 - val_loss: 0.0872
Epoch 7/10
114/114 [=====] - 0s 184us/step - loss: 0.0576 - acc: 1.0000 - val_loss: 0.0872
Epoch 8/10
114/114 [=====] - 0s 220us/step - loss: 0.0365 - acc: 1.0000 - val_loss: 0.0872
Epoch 9/10
114/114 [=====] - 0s 278us/step - loss: 0.0236 - acc: 1.0000 - val_loss: 0.0872
Epoch 10/10
114/114 [=====] - 0s 231us/step - loss: 0.0165 - acc: 1.0000 - val_loss: 0.0872
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.7516 - acc: 0.2105 - val_loss: 0.0872
Epoch 2/10
114/114 [=====] - 0s 235us/step - loss: 0.6624 - acc: 0.7632 - val_loss: 0.0872
Epoch 3/10
114/114 [=====] - 0s 268us/step - loss: 0.3393 - acc: 0.9211 - val_loss: 0.0872
Epoch 4/10
114/114 [=====] - 0s 236us/step - loss: 0.1607 - acc: 1.0000 - val_loss: 0.0872
Epoch 5/10
114/114 [=====] - 0s 184us/step - loss: 0.0775 - acc: 1.0000 - val_loss: 0.0872
Epoch 6/10
114/114 [=====] - 0s 249us/step - loss: 0.0434 - acc: 1.0000 - val_loss: 0.0872
Epoch 7/10
114/114 [=====] - 0s 265us/step - loss: 0.0279 - acc: 1.0000 - val_loss: 0.0872
Epoch 8/10
114/114 [=====] - 0s 271us/step - loss: 0.0193 - acc: 1.0000 - val_loss: 0.0872
Epoch 9/10
114/114 [=====] - 0s 216us/step - loss: 0.0135 - acc: 1.0000 - val_loss: 0.0872
Epoch 10/10
114/114 [=====] - 0s 330us/step - loss: 0.0095 - acc: 1.0000 - val_loss: 0.0872
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.6753 - acc: 0.2368 - val_loss: 0.0872
Epoch 2/10
114/114 [=====] - 0s 204us/step - loss: 0.5428 - acc: 0.7807 - val_loss: 0.0872

```



```

Epoch 3/10
114/114 [=====] - 0s 213us/step - loss: 0.2207 - acc: 0.9912 - val_loss: 0.734893
Epoch 4/10
114/114 [=====] - 0s 174us/step - loss: 0.0867 - acc: 1.0000 - val_loss: 0.734893
Epoch 5/10
114/114 [=====] - 0s 214us/step - loss: 0.0419 - acc: 1.0000 - val_loss: 0.734893
Epoch 6/10
114/114 [=====] - 0s 195us/step - loss: 0.0247 - acc: 1.0000 - val_loss: 0.734893
Epoch 7/10
114/114 [=====] - 0s 197us/step - loss: 0.0160 - acc: 1.0000 - val_loss: 0.734893
Epoch 8/10
114/114 [=====] - 0s 214us/step - loss: 0.0107 - acc: 1.0000 - val_loss: 0.734893
Epoch 9/10
114/114 [=====] - 0s 213us/step - loss: 0.0073 - acc: 1.0000 - val_loss: 0.734893
Epoch 10/10
114/114 [=====] - 0s 736us/step - loss: 0.0050 - acc: 1.0000 - val_loss: 0.734893
Train on 114 samples, validate on 1026 samples
Epoch 1/10
114/114 [=====] - 0s 2ms/step - loss: 1.8982 - acc: 0.1491 - val_loss: 0.734893
Epoch 2/10
114/114 [=====] - 0s 216us/step - loss: 0.6363 - acc: 0.7895 - val_loss: 0.734893
Epoch 3/10
114/114 [=====] - 0s 162us/step - loss: 0.3003 - acc: 0.9123 - val_loss: 0.734893
Epoch 4/10
114/114 [=====] - 0s 205us/step - loss: 0.1106 - acc: 0.9912 - val_loss: 0.734893
Epoch 5/10
114/114 [=====] - 0s 161us/step - loss: 0.0424 - acc: 1.0000 - val_loss: 0.734893
Epoch 6/10
114/114 [=====] - 0s 199us/step - loss: 0.0234 - acc: 1.0000 - val_loss: 0.734893
Epoch 7/10
114/114 [=====] - 0s 195us/step - loss: 0.0150 - acc: 1.0000 - val_loss: 0.734893
Epoch 8/10
114/114 [=====] - 0s 203us/step - loss: 0.0104 - acc: 1.0000 - val_loss: 0.734893
Epoch 9/10
114/114 [=====] - 0s 239us/step - loss: 0.0073 - acc: 1.0000 - val_loss: 0.734893
Epoch 10/10
114/114 [=====] - 0s 239us/step - loss: 0.0052 - acc: 1.0000 - val_loss: 0.734893
The best npc is 60, and the best nnode is 200.
The best validation accuracy is 0.734893.

```

In [43]: *#CNN*

```

Xtr_cnn = X_train.astype("float32")/255
Xts_cnn = X_test.astype("float32")/255
Xtr_cnn = np.reshape(Xtr_cnn, (len(Xtr_cnn),h,w,1))
Xts_cnn = np.reshape(Xts_cnn, (len(X)-len(Xtr_cnn),h,w,1))
K.clear_session()
model = Sequential()

```

```

model.add(Conv2D(8, (5, 5),
                padding='valid',
                input_shape=Xtr_cnn.shape[1:],
                activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(8, (5, 5), padding='valid', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(200, activation='relu'))
model.add(Dense(nout, activation='softmax'))
opt = optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=1e-08, decay=0.0)
# Let's train the model using Adam
model.compile(loss='sparse_categorical_crossentropy',
              optimizer=opt,
              metrics=['accuracy'])
hist_basic = model.fit(Xtr_cnn, y_train, batch_size=100, epochs=40,
                      validation_data=(Xts_cnn, y_test), shuffle=True)
print("The accuracy on validation set is:")
print(hist_basic.history['val_acc'])

```

Train on 114 samples, validate on 1026 samples

Epoch 1/40

114/114 [=====] - 1s 10ms/step - loss: 1.5895 - acc: 0.1579 - val_loss

Epoch 2/40

114/114 [=====] - 1s 6ms/step - loss: 1.4490 - acc: 0.4649 - val_loss

Epoch 3/40

114/114 [=====] - 1s 6ms/step - loss: 1.4300 - acc: 0.4649 - val_loss

Epoch 4/40

114/114 [=====] - 1s 6ms/step - loss: 1.4118 - acc: 0.4649 - val_loss

Epoch 5/40

114/114 [=====] - 1s 7ms/step - loss: 1.4072 - acc: 0.4649 - val_loss

Epoch 6/40

114/114 [=====] - 1s 6ms/step - loss: 1.3997 - acc: 0.4649 - val_loss

Epoch 7/40

114/114 [=====] - 1s 6ms/step - loss: 1.3937 - acc: 0.4649 - val_loss

Epoch 8/40

114/114 [=====] - 1s 6ms/step - loss: 1.3846 - acc: 0.4649 - val_loss

Epoch 9/40

114/114 [=====] - 1s 6ms/step - loss: 1.3803 - acc: 0.4649 - val_loss

Epoch 10/40

114/114 [=====] - 1s 5ms/step - loss: 1.3776 - acc: 0.4649 - val_loss

Epoch 11/40

114/114 [=====] - 1s 5ms/step - loss: 1.3877 - acc: 0.4649 - val_loss

Epoch 12/40

114/114 [=====] - 1s 5ms/step - loss: 1.3900 - acc: 0.4649 - val_loss

Epoch 13/40

114/114 [=====] - 1s 5ms/step - loss: 1.3607 - acc: 0.4649 - val_loss

Epoch 14/40

114/114 [=====] - 1s 5ms/step - loss: 1.3624 - acc: 0.4649 - val_loss
 Epoch 15/40
 114/114 [=====] - 1s 6ms/step - loss: 1.3707 - acc: 0.4649 - val_loss
 Epoch 16/40
 114/114 [=====] - 1s 5ms/step - loss: 1.3646 - acc: 0.4737 - val_loss
 Epoch 17/40
 114/114 [=====] - 1s 6ms/step - loss: 1.3531 - acc: 0.4649 - val_loss
 Epoch 18/40
 114/114 [=====] - 1s 5ms/step - loss: 1.3442 - acc: 0.4649 - val_loss
 Epoch 19/40
 114/114 [=====] - 1s 5ms/step - loss: 1.3362 - acc: 0.4649 - val_loss
 Epoch 20/40
 114/114 [=====] - 1s 5ms/step - loss: 1.3246 - acc: 0.4649 - val_loss
 Epoch 21/40
 114/114 [=====] - 1s 7ms/step - loss: 1.3189 - acc: 0.4649 - val_loss
 Epoch 22/40
 114/114 [=====] - 1s 8ms/step - loss: 1.3157 - acc: 0.4649 - val_loss
 Epoch 23/40
 114/114 [=====] - 1s 8ms/step - loss: 1.3067 - acc: 0.4649 - val_loss
 Epoch 24/40
 114/114 [=====] - 1s 7ms/step - loss: 1.2926 - acc: 0.4649 - val_loss
 Epoch 25/40
 114/114 [=====] - 1s 7ms/step - loss: 1.2768 - acc: 0.4649 - val_loss
 Epoch 26/40
 114/114 [=====] - 1s 7ms/step - loss: 1.2780 - acc: 0.4912 - val_loss
 Epoch 27/40
 114/114 [=====] - 1s 9ms/step - loss: 1.2625 - acc: 0.5439 - val_loss
 Epoch 28/40
 114/114 [=====] - 1s 8ms/step - loss: 1.2444 - acc: 0.4825 - val_loss
 Epoch 29/40
 114/114 [=====] - 1s 7ms/step - loss: 1.2335 - acc: 0.4649 - val_loss
 Epoch 30/40
 114/114 [=====] - 1s 7ms/step - loss: 1.2311 - acc: 0.4649 - val_loss
 Epoch 31/40
 114/114 [=====] - 1s 7ms/step - loss: 1.2195 - acc: 0.4649 - val_loss
 Epoch 32/40
 114/114 [=====] - 1s 8ms/step - loss: 1.2071 - acc: 0.4649 - val_loss
 Epoch 33/40
 114/114 [=====] - 1s 8ms/step - loss: 1.1675 - acc: 0.4737 - val_loss
 Epoch 34/40
 114/114 [=====] - 1s 7ms/step - loss: 1.1577 - acc: 0.5789 - val_loss
 Epoch 35/40
 114/114 [=====] - 1s 7ms/step - loss: 1.1555 - acc: 0.5877 - val_loss
 Epoch 36/40
 114/114 [=====] - 1s 7ms/step - loss: 1.1208 - acc: 0.5877 - val_loss
 Epoch 37/40
 114/114 [=====] - 1s 7ms/step - loss: 1.0941 - acc: 0.5439 - val_loss
 Epoch 38/40

```
114/114 [=====] - 1s 7ms/step - loss: 1.0740 - acc: 0.5439 - val_loss
Epoch 39/40
114/114 [=====] - 1s 7ms/step - loss: 1.0460 - acc: 0.5789 - val_loss
Epoch 40/40
114/114 [=====] - 1s 8ms/step - loss: 1.0384 - acc: 0.6404 - val_loss
The accuracy on validation set is:
[0.4649122842455003, 0.4649122842455003, 0.4649122842455003, 0.4649122842455003, 0.4649122842455003]
```

Q: How does CNN compare with PCA+NN with the small training set? Why?

A: The validation accuracy CNN gets is much smaller than that of PCA+NN. It probably because of its smaller training data set.