Homework 2: Eigenfaces

Robert Grant
2009-09-13

1 The Assignment

In this homework, I used Principal Components Analysis (PCA) to classify face images from three datasets. Principal Components analysis involves unrolling each face image into a vector, and then considering each pixel as belonging to a dimension in that vector. Vectors for all faces are then concatenated into a $D \times N$ matrix A which I will call a *gallery matrix*, where D is the number of dimensions and N is the number of images. Then, one finds the eigenvector decomposition of the covariance of A; this would be very computationally intensive considering the size of this matrix, but there is a trick for using PCA with high-dimensional data, as outlined in class and in PRML [1]. The eigenvector matrix V that one can compute with this trick is then sorted in descending order by eigenvalue, meaning the eigenvector in the direction of greatest variance is first. V can then be seen as a set of lower-dimensional basis vectors which represent the most significant components of the original matrix A.

To use V for face classification, I projected both a set of training data and a set of test data onto V. Then, for each point in the test set, I implemented a nearest neighbor algorithm to match the point to its most similar point in the training set. Then, the test point was classified however the training point was classified.

For the above algorithm to work, it required human-generated classification labels for all of the points in the training set. To accurately assess the classification algorithm's performance, all of the test data needed human-generated labels as well, for comparison. To assess the algorithm's performance, I compared its classification of a test set to the human-labelled "true" values and computed algorithm's error rate.

2 Implementation

My implementation is contained in four files: `hw2Main.m`, `hw2LoadPictureFiles.m`, `hw2FindEigenfaces.m`, and `hw2ClassifyEigenfaces.m`. `hw2Main.m` is the top level script file which calls the other functions, varies parameters, and generates plots. `hw2LoadPictureFiles.m`, when configured with the correct file paths, loads in the picture files for all the datasets, formats them into gallery matrices, attaches some relevant metadata to them, and returns them in a cell array. `hw2FindEigenfaces.m` is where PCA is actually implemented according the specifications in the homework document. `hw2ClassifyEigenfaces.m` is where the data is separated into training and test sets, the nearest-neighbor algorithm is applied, and error rates for each classification type are computed.

Figure 1 is a call graph showing the relationship between the files, where an arrow from one file to another means "calls". Please see the comments at the top of each file for more detailed instructions on calling these functions. These comments are also accessible via the usual `help function_name`.

3 Results

As my first test, I simply plotted the mean images and a few eigenvectors. The mean image of each dataset along with the first eigenvector+mean is shown in Figures 2, 3, and 4.

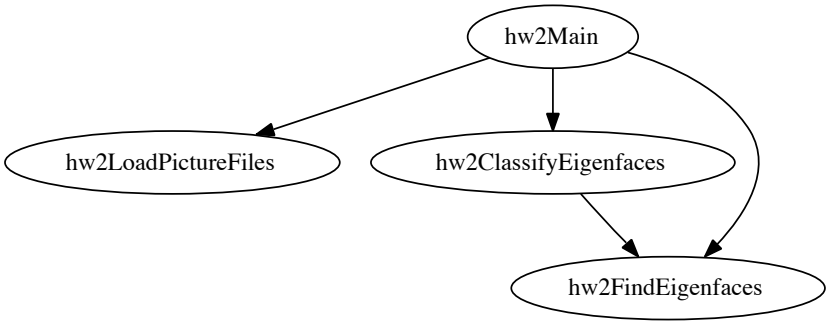

Figure 1: Implementation call graph

Figure 2: Mean and first eigenvector of class08 dataset

Figure 3: Mean and first eigenvector of class09 dataset


Figure 4: Mean and first eigenvector of yale dataset

As my main experiment, I varied the percentage of data used for training and plotted the error rate. Before doing this, I hand labelled all of the pictures in each of the datasets for three characteristics:

1. Is the subject smiling?
2. Is the subject wearing glasses?
3. Is the subject male?

The data taken for training was taken right from the beginning of the dataset. In other words, it was not randomized, nor was it specialized for its use on the above variables. As seen from Figures 5, 6, and 7, the best amount for training seems to be 30% to 40%. The yale dataset seems to have consistenly lower error rates than the other two, likely because it contains so many more images. Also, the *Male* variable seems to be the easiest to discriminate, which the *Glasses* variable seems to be the most difficult.

4 Other Notes

The class09 dataset had one oddly sized picture, which I simply discarded.

References

[1] Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2007. Section 12.1.4, pp. 569-570.

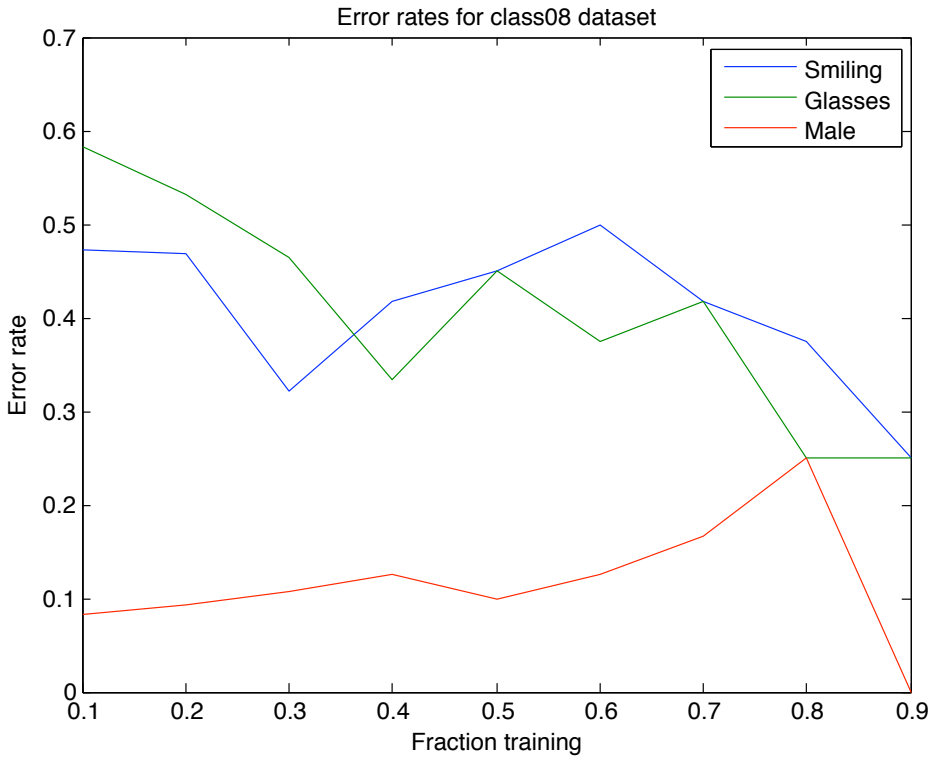

Figure 5: Error rate on class08 dataset

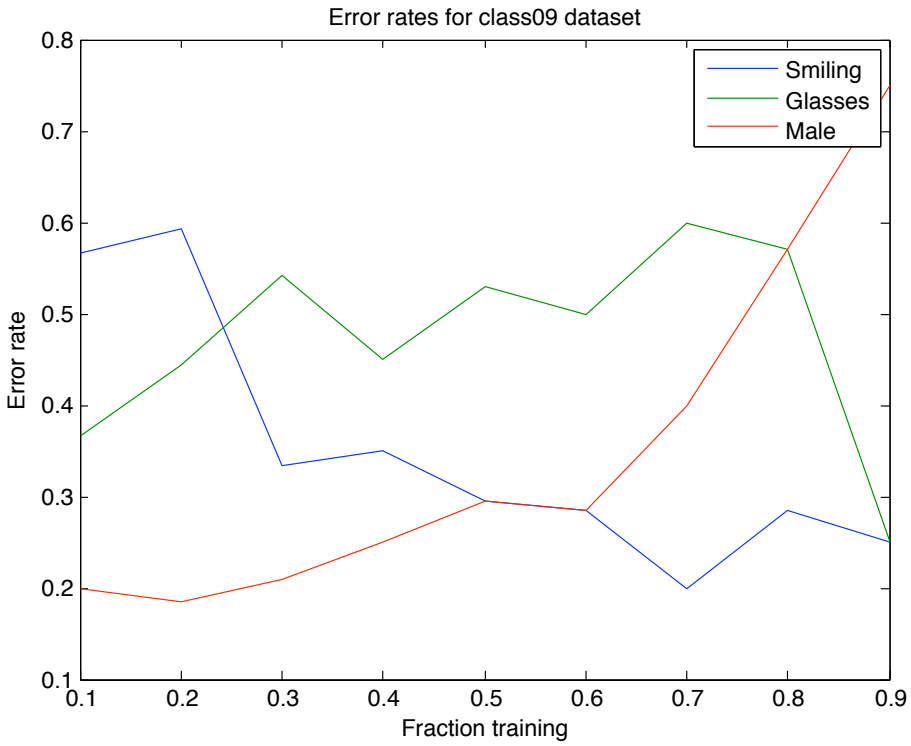

Figure 6: Error rate on class09 dataset

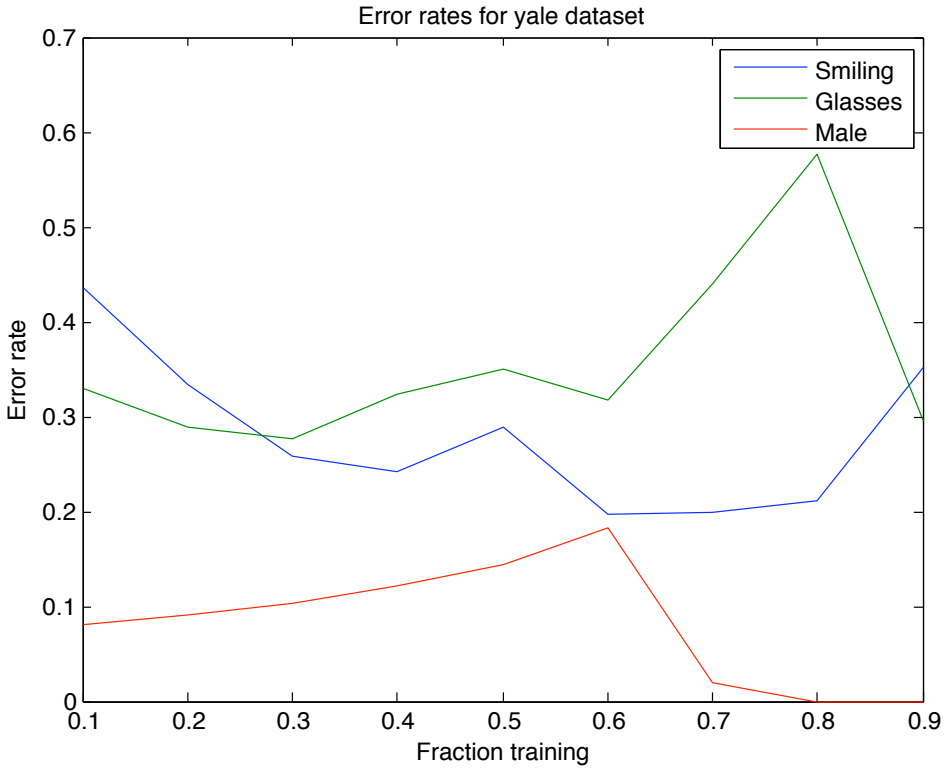

Figure 7: Error rate on yale dataset