# COMPUTER VISION ASSIGNMENT 5

JUSTIN DE WITT* (21663904)

6 October 2022

## 1 PCA FEATURE VECTORS AND K-NN CLASSIFICATION

In this problem we would like to build a simple facial recognition system, using PCA feature vectors. To implement the system, we are using a *Georgia Tech* face database, composed of 750 photos, taken of 50 individuals.

### 1.1 Image Standardization

To determine the PCA feature vectors, we must first regularize the data. That is, we must guarantee that all images in the dataset are equal in size. There are two obvious approaches:

1. We can consider cropping each image about the image center. To implement this method, we must first determine the smallest image in the dataset. It was discovered that the smallest image in the dataset contains 156 rows, and 111 columns. It therefore makes sense crop a 154x110 rectangle out of the center of each image. This method may however cause information loss. It may be the case that the entirety of the face may not fit into the cropped result.
2. The second method to consider is strictly resizing each image. This method will not preserve the aspect ratio of each image. If some images contain a different row-to-column ratio, the faces may appear distorted if strictly resized to fit a $m$x$n$ frame.

There are drawbacks associated with each of the mentioned methods. For example, consider the images shown in the following figures.

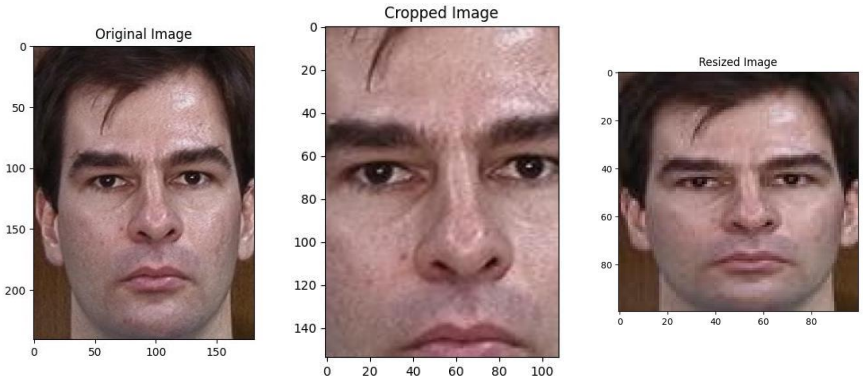



**Figure 1:** A side-by-side view of the unadjusted image, as well as the regularized version, using both methods.

As you can see in the figure above, the cropped version of the image contains information loss. We do not have the option to crop the image to a larger size, because this would imply cropping to a size which exceeds that of the smallest image. We therefore need to work with this information loss. Furthermore, we observe that the man's face appears wider in the resized image. This is because the image was re-sampled to fit a square. I have experimented with different aspect ratios, and did not observe any meaningful improvements. I have therefore simply opted to use a 100x100 square for this method. All of the images were subsequently regularized using each of the respective methods.

## 1.2 Sampling of the Hidden Test Set

To eventually assess the model's classification success, a set of regularized images must be set aside. For this, we randomly sample 5 images from each of the 50 photographed individuals. It is important that we remove these 250 images from the set. We do not allow these images to remain in the set used to train the model, otherwise our classification accuracy will be skewed.

## 1.3 Obtaining an Image Vector, the Average Vector, and a Matrix of Basis Vectors

In obtaining the values discussed in this subsection, we sample five photos (from the remaining 10) at random, for each of the 50 individuals in our dataset. We use this set of 250 photos for the derivation of the following values.

### *1.3.1 An Image Vector*

We need to devise a method to transform an image, into a single Mx1 dimensional vector. To obtain this vector, we simply stack the columns of the image, into a single vector. Note that the image is in colour, therefore each pixel does not represent a single value, but rather three values. We subsequently also unstring each pixel into a vertical stack of its three coloured components. The result of a "unstrung" MxN dimensional image, is a $(3 * M * N)$x1 dimensional vector. We begin by "unstringing" each regularized RGB image into its corresponding image vector. When we refer to *each*, we are referring to the remaining 500 images, as we have removed 250 images to form the evaluation set.

### *1.3.2 The Average Vector*

Now that a method has been devised to decompose an image into its corresponding image vector, we define the average vector as

$$\underline{\mathbf{a}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{f}_i$$

where $\mathbf{f}_i$ is the image vector corresponding to image $i$.

## 1.4 A Discussion as to the Relevance of the Average Vector

If you consider a hypothetical high dimensional coordinate system, where each image vector is drawn from the origin, it does not seem far-fetched to argue that the set of image vectors are contained in a high dimensional cloud, somewhere on this coordinate system. Recall that applying transformations (associated with matrix multiplication) is often much more simple, if the points of interest are centered about the origin. Consider applying a rotation for example. It is trivial to define a matrix which rotates about the origin, however defining a transformation which rotates around a point in M dimensional space is complex. Rotating a shape about

a point in M dimensional space need not be complex however, we simply translate to the origin, apply the trivial transformation, and translate the shape back to its original location. It is therefore useful to center the data of interest about the origin, so that a single defined transformation can be applied. Otherwise, a complex (and unique) transformation need be devised depending on the coordinate location of the field of interest. We can easily center our hypothetical vector cloud about the origin, by simply subtracting the *average* vector, from each vector in the cloud. In this context, the *average* vector is interpreted as a center of gravity, where all elements of the cloud contribute an equal weight. We show the image representation of this *average* vector in a proceeding section.

### 1.5 Determining the Basis Vector Set

As discussed in the previous subsection, we should center our high-dimensional datapoints about the origin. That is to say, we need to subtract the mean vector from each of the image vectors. This subtraction yields a new vector, $\mathbf{x}_i$, which matches the dimension of the image vector, however it has been translated towards the origin. To determine our basis vector set, we begin by packing all of our 250 $\mathbf{x}_i$ vectors into the columns of a matrix. We thereafter scale this matrix by constant factor, $\sqrt{250}$ in this case. We call this matrix X, and is size $(M * N * 3)$x250, where each regularized image is MxN. We now seek to obtain a basis which spans the column space of X. The spectral values associated with the spectral value decomposition (SVD) of X is helpful here. We plot the first 250 spectral values in the following figure.

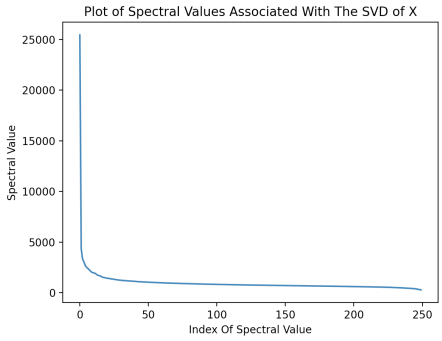



**Figure 2:** A plot of the 250 most significant spectral values associated with the SVD of X.

We are interested in the retaining the principle components in order of significance. Spectral values are directly related to eigenvalues, which give the *amount of variance* associated with each principle component. We wish to retain the components which explain most of the total variance, since variance is associated with information. If we consider the plot, it seems that most of the variance is explained by using the first 20 principle components, as thereafter each remaining principle component is associated with very little information. We therefore estimate that using approximately 20 columns of the U matrix (obtained from the SVD of X) span a sufficient low-dimensional subspace.

### 1.6 Dimensionality Reduction

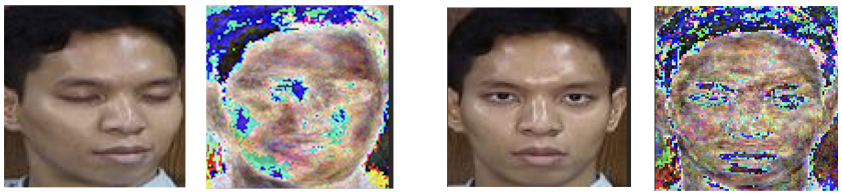
We can now solve for the linear combination of basis vectors, coordinates, which best approximate each mean-centered image vector. That is, we can easily obtain $\underline{\mathbf{y}}$:

$$\begin{aligned} U_\alpha \underline{\mathbf{y}_i} &= \underline{\mathbf{x}_i} \\ \underline{\mathbf{y}_i} &= U_\alpha^T(\underline{\mathbf{x}_i}) \end{aligned} \tag{1}$$

because the transpose of $U_\alpha$ is its inverse (orthogonal columns). In this equation, $\underline{x_i}$ represents any mean-centered image vector. The resulting coordinate vector $\underline{y_i}$ represent the combinations of the basis vectors needed to (approximately) estimate our mean-centered vector $\underline{x_i}$. Trivially, the dimension of our coordinate vector matches the quantity of vectors forming our basis, and our estimation accuracy is increased if we *increase* the size of the basis, after-all, this allows for the explanation of more variance. **However**, this is not to say that increasing the basis size will lead to better classification, more on that to come.

### 1.7 Reconstructed Image Estimations

We can easily reconstruct an image estimation obtaining our $\underline{x_i}$ estimate. We obtain this estimate by summing the coefficient of each basis vectors contribution (coordinate), with that basis vector. The resulting vector sum, is our $\underline{x_i}$ estimate. Simply under the mean-centering by adding the average vector $\underline{a}$, and "unstring" the result. A few figures of different basis sizes are shown below.



Basis Width: 40 Basis Width: 106

Basis Width: 172 Basis Width: 240

**Figure 3:** A comparison of image reconstruction using various basis sizes. The reconstructed estimation is shown to the right of the original. Note the direct correlation between basis size, and reconstruction accuracy. Once again, this later shown to not be directly correlated to eventual classification accuracy.

### 1.8 Eigen-face Interpretation

The reconstruction of our approximated mean-centered vectors $\underline{x_i}$, and the image estimations thereafter, poses an interesting thought. What if the coordinate used in the reconstruction were $[0_1, 0_2, ..., 1_k ..., 0_\alpha]$, where $\alpha$ is the number of vectors composing our basis. This is equivalent to asking the question: *What happens if we choose one of our basis vectors* ***as*** *a mean-centered estimate?* We can map each basis vector into an image (hopefully resembling a face). Just a linear combination of our basis vectors represent a low dimensional representation of our original image, we can likewise interpret this finding as a combination of "face components" estimating our mean-centered vector. This concept is referred to as *eigenfaces*. We show an example of the 4 most significant eigenfaces, and the respective linear combinations below.

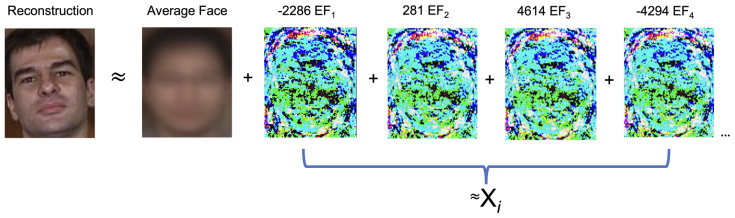



**Figure 4:** The four most significant eigenfaces, along with their respective coordinate contributions. Here a basis width of 240 was used, because it leads to the most pretty illustrations. We will determine later that a larger basis width does not yield optimal classification.

## 1.9 Classifying Feature Vectors Using K-NN

We now convert all the images in the dataset (including test data) to feature vectors. That is, we obtain the respective linear combination of basis vectors that best estimates the mean-centered image vector $\underline{x_i}$. Now that both the training set (10 images per person) and test set (5 images per person) have been converted to feature vectors, we need to label the sets respectively. That is, we create two arrays, corresponding to the length of the test set, and training set respectively. Each index of the array, contains a numeric value associated with the "number" of the individual, the classifier will attempt to guess this number correctly.

### *1.9.1 K-NN Classification*

The K-nearest neighbor algorithm works by finding the $k$ points in the sample set, that are nearest to a query point. Different definitions to define distance are used, but for the purpose of this paper, we use Euclidean distance. Once the set of $k$ datapoints is obtained, the query point is classified as the majority class in the set. In the case of ties (even neighborhood size), *Scikit-Learn* uses a "first class to appear" approach, however the classes are returned in sorted order. Therefore, the final classification corresponds to the class of nearest distance.

## 1.10 Accuracy of K-NN for Various K

Here we plot the accuracy of the K-NN classification for various neighborhood sizes, and basis sizes. We obtain a plot for both the standardization methods discussed in section 1.1. Consider the following figures.

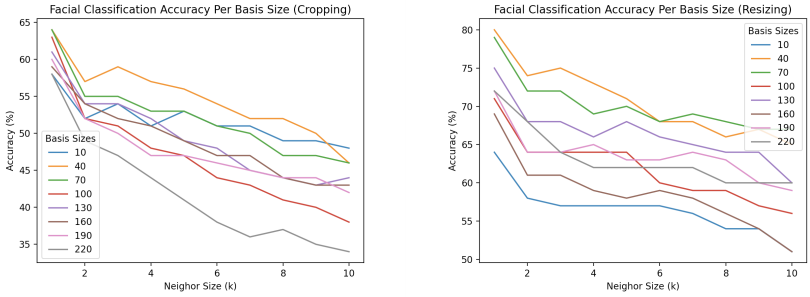



**Figure 5:** Classification accuracy using cropping and resizing as the standardization technique.

Based on the results from figure 5, we determine that a low neighborhood size (usually 1) and a basis width of approximately 25-40 yields the best classification accuracy.