Louis Sharrock Imperial College London

Principal Components Analysis

1. Summary

This project investigates the use of dimensionality reduction via principal component analysis (PCA) within a classification algorithm for face recognition. We consider a data set consisting of training and test images of faces, each of which is 112 pixels tall and 92 pixels wide. By computing a PCA basis we construct a lower dimensional representation of these images, which we then use within a nearest neighbour classifier.

2. Preliminaries

2.1. The Data

Our data consists of of 400 images of faces. We split this into a training set (320 images) and a testing set (80 images). Each of the images is 112 pixels tall and 92 pixels wide, and represented by a $112 \times 92 = 10304$ dimensional column vector $\mathbf{x}_i$. We store our two data sets as matrices, with images represented by the respective row vectors:

$$[X_{\text{train}}]_{320\times 10304} = (\mathbf{x}_1, \dots, \mathbf{x}_{320})^T \quad , \quad [X_{\text{test}}]_{80\times 10304} = (\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_{80})^T \tag{1}$$

We provide illustrative plots of several of the faces in the training set, as well as the 'average face', defined as $\mathbf{m} = \frac{1}{320}\sum_{i=1}^{320} \mathbf{x}_i$, in Figure 1.

(a) Training Image 21 (b) Training Image 108 (c) Training Image 319. (d) Average Face

Figure 1: Plots of three images from the training set, alongside the 'average face'.

3. Principal Components Analysis

3.1. Outline

Broadly speaking, Principal Components Analysis (PCA) seeks the set of M orthonormal basis vectors, say $\{\mathbf{v}_n\}_{n=1}^M$, which in some sense best describes the distribution of the data. More precisely, the i^{th} basis vector $\mathbf{v}_i$ is chosen to maximise the quantity

$$\lambda_i = \frac{1}{M}\sum_{n=1}^{M}(\mathbf{v}_i^T \mathbf{y}_n)^2 \tag{2}$$

where $\mathbf{y}_i = \mathbf{x}_i - \mathbf{m}$ denote the mean-centered training inputs.

3.2. Computation

To compute the $\mathbf{v}_i$, we use the fact that the PCA basis of size M precisely corresponds to the eigenvectors associated with the M largest eigenvalues of the covariance matrix

$$S = \frac{1}{320}\sum_{i=1}^{320} \mathbf{y}_n \mathbf{y}_n^T = Y^T Y \tag{3}$$

where we have defined $Y_{\{320\times 10304\}} = (\mathbf{y_1}, \dots, \mathbf{y}_{320})^T$. Given that S has dimensions 10304×10304, it is somewhat time consuming to naively compute its eigenvectors directly. We thus recall the following proposition.

Proposition. *Let a real matrix A have dimension $n \times p$, where $n \leq p$. Then the number of eigenvectors with nonzero eigenvalue of $A^T A_{\{p\times p\}}$ is no greater than the number of eigenvectors with nonzero eigenvalue of $AA^T_{\{n\times n\}}$.*

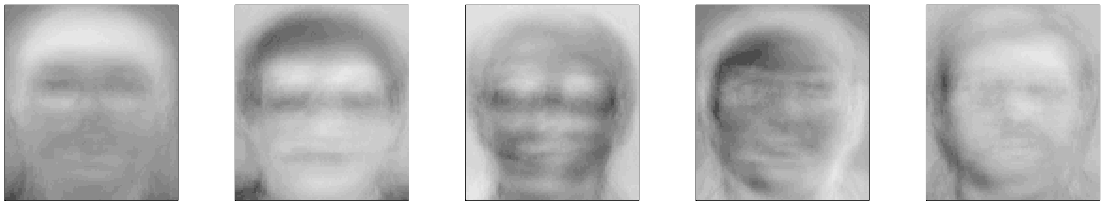
Since eigenvectors with zero eigenvalues are removable[1], it suffices to solve for the eigenvectors of the 320×320 dimensional matrix YY^T. Indeed, supposing $\mathbf{u}_i$ is a normalised eigenvector of YY^T with eigenvalue λ_i, we have

$$YY^T\mathbf{u}_i = \lambda_i \mathbf{u}_i \implies Y^T Y(Y^T\mathbf{u}_i) = \lambda_i(Y^T\mathbf{u}_i) \tag{4}$$

so that $Y^T\mathbf{u}_i$ is an eigenvector of $S = Y^TY$, also with eigenvalue λ_i. We can thus use the following algorithm to determine the PCA basis $\{\mathbf{v}_i\}_{i=1}^M$ of size M for training data X:

(i) mean center $X = (\mathbf{x}_1, \dots, \mathbf{x}_{320})^T$, to form $Y = (\mathbf{y_1}, \dots, \mathbf{y}_{320})^T = (\mathbf{x}_1 - \mathbf{m}, \dots, \mathbf{x}_{320} - \mathbf{m})^T$;
(ii) construct $YY^T_{\{320\times 320\}}$;
(iii) compute the normalised eigenvectors $\{\mathbf{u}_i\}_{i=1}^M$ of YY^T corresponding to its M largest eigenvalues $\{\lambda_i\}_{i=1}^M$;
(iv) form the normalised PCA basis vectors $\{\mathbf{v}_i\}_{i=1}^M = \dfrac{Y^T\mathbf{u}_i}{||Y^T\mathbf{u}_i||}$, with eigenvalues $\{\lambda_i\}_{i=1}^M$.

We implement this procedure in `R`, using the in-built function `eigen` to determine the required eigenvectors of YY^T. Using this function, it is simple to plot the first 5 eigenfaces of the training set (Figure 2).



(a) Eigenface 1 (b) Eigenface 2 (c) Eigenface 3 (d) Eigenface 4 (e) Eigenface 5

Figure 2: Plots of the first 5 eigenfaces of the training set.

[1] They correspond to zero variability of a face in the direction of the corresponding eigenvector.

3.3. Application

We now restrict our attention to a particular training face, $\mathbf{x}_{15}$, and project it into a PCA basis for dimensions $M = 5, 10, 50$. Defining $V_M = (\mathbf{v}_1, \ldots, \mathbf{v}_M)$ as the matrix with columns corresponding to the ordered basis vectors of the M-dimensional PCA basis, the required M-dimensional (column) vector $\mathbf{w}_M$ in 'face space' is determined by

$$\mathbf{w}_M^T = (\mathbf{x}_{15} - \mathbf{m})^T V_M \tag{5}$$

We note that the subscript M is not an index, but denotes the dependence of the projection vector on the dimension of the chosen PCA basis. To plot the results of our projection, it remains to transform this projection back to a 10304 dimensional (column) vector $\hat{\mathbf{x}}_{15}$ in the 'image space':[2]

$$\hat{\mathbf{x}}_{15,M}^T = \mathbf{w}_M^T V_M^T + \mathbf{m}^T = (\mathbf{x}_{15} - \mathbf{m})^T V_M V_M^T + \mathbf{m}^T \tag{6}$$

The resulting plots, alongside the original image, are displayed in Figure 3. We note, as expected, that the 'resolution' or 'clarity' of the images increases as a function of the number of basis vectors.

(a) Original Image (b) $M = 5$ (c) $M = 10$ (d) $M = 50$

Figure 3: Plots of three M-dimensional approximations of training image 15 ($M = 5, 10, 50$), alongside the original image.

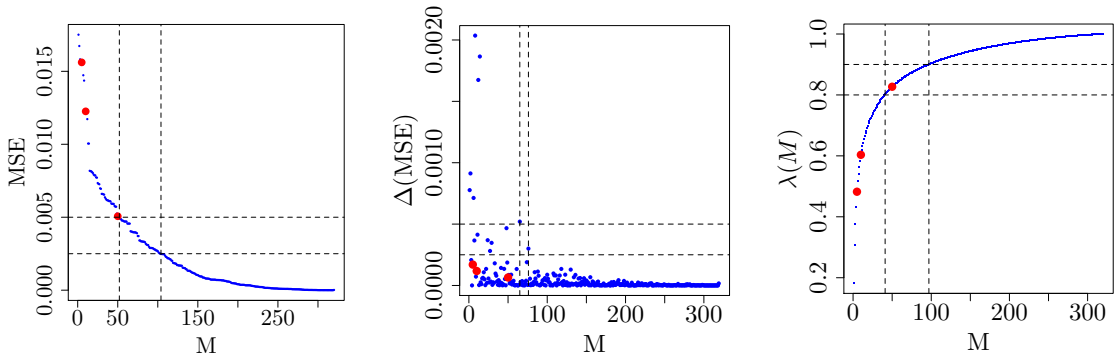
3.4. The 'Optimal' Basis

By computing the squared difference between each coordinate of the original face vector, and each coordinate of its approximation as defined by equation (6), we can now compute the mean squared error (MSE), of each of our lower dimensional approximations, defined as

$$\text{MSE}_M = \frac{1}{10304} \sum_{i=1}^{10304} \left(\{\mathbf{x}_{15,M}\}_i - \{\hat{\mathbf{x}}_{15,M}\}_i \right)^2 \tag{7}$$

A plot of the MSE as a function of the dimensionality M is provided in Figure 4(a), for dimensions $M = 1, \ldots, 320$. The values of the MSE at the specified dimensions of interest, $M = 5, 10, 50$, are highlighted in red. From this Figure, it is difficult to say with confidence that there is a single clear point at which we achieve a 'good approximation'. Indeed this term is, by definition, somewhat subjective. In this context, we consider a number of distinct approaches which attempt to determine an 'optimal' dimensionality.

[2] We remark that decision to make the subject of each of these equations a row vector (i.e. to include the transpose) is arbitrary, and is done simply to maintain consistency with the representation of our images in R as rows of a matrix.



(a) MSE(M). (b) $\Delta(MSE)(M)$. (c) $\lambda(M)$.

Figure 4: Plots of three measures for assessing the optimal choice of dimension M, for training image 15.

Clearly it is desirable to achieve a suitably low value of the mean squared error between our chosen approximation and the original image. As such, the first, and perhaps most naive approach to specifying a value of M is simply to set a threshold value, say ε, for the MSE, and choose the minimum value of M such that $\text{MSE}_M < \varepsilon$. The minimum such value is chosen in the interest of parsimony.

An alternative approach is to consider the rate at which the MSE decreases as a function of M, which corresponds to the magnitude of the 1-step finite differences in our graph of $f(M) = \text{MSE}(M)$. Again, the optimal (range of) dimensions are then determined by setting a threshold value δ for $\Delta(\text{MSE})(M) = \text{MSE}(M+1) - \text{MSE}(M)$, and selecting the minimum value of M such that $\Delta(\text{MSE})(M) < \delta$ for all $m \geq M$. This approach encompasses the idea that we would like to specify a dimension which, were it to be increased any further, would not result in a significant improvement in our approximation. A plot of $\Delta(MSE)$ as a function of dimensionality is included in Figure 4(b).

A third potential approach is to require that a certain proportion of the variance in the original image should be explained by our lower dimensional approximation. Since the eigenvalue of each eigenface is in some sense a measure of the variance in the original image explained by that eigenface, in practice this corresponds to selecting a threshold value γ for the quantity

$$\lambda(M) = \frac{\sum_{i=1}^{M} \lambda_i}{\sum_{i=1}^{320} \lambda_i} \tag{8}$$

where λ_i is the eigenvalue corresponding to the i^{th} PCA basis vector. Thus $\lambda(M)$ represents the sum of the eigenvalues of the first M PCA basis vectors, as a proportion of the sum of all possible non-zero eigenvalues. In this case, we select the minimum value of M such that $\lambda(M) > \gamma$. This quantity is graphed in Figure 4(c).

Ideally, the specification of an appropriate value of M will consider each of these three approaches. On this basis, for the given image we would argue that a value of M in the approximate range $M \in [40, 100]$ is optimal. Qualitatively, the lower end of this range is supported by our 50 dimensional approximation of training image 15, as plotted in Figure 3(d), which bears a strong resemblance to the original image. More quantitatively, for this range of M, the MSE is suitably low, the rate of reduction in the MSE is insignificant for higher values of M, and a reasonable proportion of the total variance is explained. These observations are made precise in Table 1, which provides numerical values corresponding to the dashed lines in Figure 4.

	MSE		Δ(MSE)		λ	
Threshold	5×10^{-3}	2.5×10^{-3}	5×10^{-4}	2.5×10^{-4}	0.8	0.9
Dimension	52	104	66	77	41	97

TABLE 1: Dimensions corresponding to upper and lower thresholds for the three metrics of interest.

Having specified such a range of dimensions, choice of a particular value of M largely depends on our subjective preference for (i) increased parsimony or (ii) increased accuracy. However, within this scheme, our choice can be locally optimised by choosing values of M which do not correspond to local maxima of $\Delta(MSE)(M)$ (or, more particularly, that immediately follow such maxima).

4. K-Nearest Neighbours

4.1. Outline

We conclude this report by implementing a K-nearest neighbour (KNN) classifier for the face recognition data. This classifier requires as inputs a training set, training labels, test set, test labels,[3] a value of K, a dimensionality M, the (full) PCA basis corresponding to the specified training set, and a value of p, the order of the computed Minkowski distance. The details of this classifier are summarised below

(i) Project the (mean-centered) test and training sets into the M-dimensional face space.
(ii) For each face in the test set, compute and store the order p Minkowski distance between that face, and all (labelled) faces in the training set.
(iii) Sort these distances in ascending order, and assign the modal class of the training faces corresponding to the smallest K distances, as the class of the given test face.[4]
(iv) Having considered all test faces, compute and return the 'classification rate' by applying the following function to the vector of true labels, and vector of labels as determined by our algorithm.

4.2. Analysis

We begin our investigation of the use of this classifier on the face recognition data-set by applying it for various values of M, K and p. Though it may be tempting to restrict our attention to $M \in [40, 100]$ on the basis of our preceeding analysis, we will continue at this point to consider a wider range of values. Indeed, if we are purely interested in classification, our conclusions regarding optimal choice(s) of M for the KNN classifier may well differ from the previously specified range. We would like to determine (whether there exists) an optimal configuration of the K-nearest neighbours algorithm for this particular application.

4.2.1 Specific Reccomendations

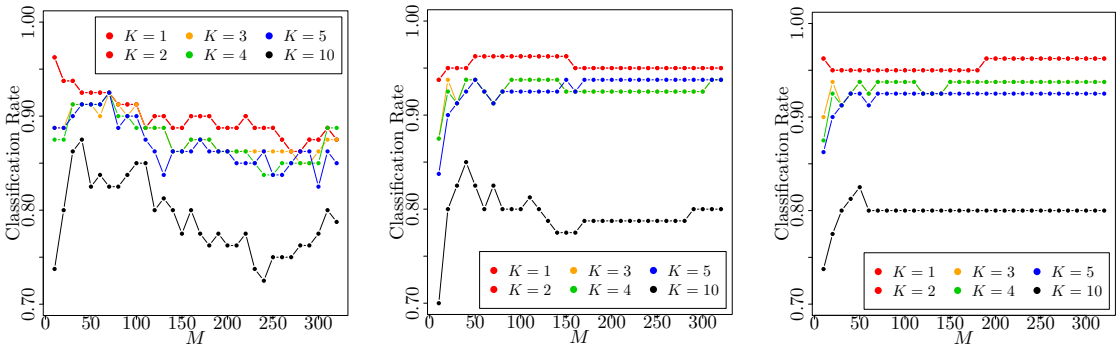
Initial investigations, a representative summary of which are recorded in Figures 5(a)-(c), strongly indicate the optimality of the choice $K = 1$ for *this* face recognition data-set, particularly in the

[3] We should note that the implementation of the classifier does not in fact depend on the test labels. However, we choose to provide as output the classification rate, which clearly does depend on these values.

[4] In the case of a tie between two classes, iteratively reduce the K nearest points by 1 until the modal class is unique.

cases $p = 2$ and $p = 3$.[5] The former of these two values corresponds to the Euclidean distance metric. In these two cases, the classification rate of our algorithm is uniquely maximised for any given value of M by this choice of K.

The case $p = 1$, which corresponds to the so-called 'Manhattan' distance metric, is less unambiguous, yet it remains generally true that the classification rate is decreasing as a function of K, for each M. In fact we can go further, noting that $K = 1$ is only outperformed by another choice of K ($K = 4$) if we use the maximal PCA basis of dimension $M = 320$. Even in this instance, the difference in performance is marginal. In this context, we are justified in restricting our attention to the case $K = 1$ for the purpose of further investigation of the performance of our classifier on the face recognition dataset.



(a) $p = 1$. (b) $p = 2$. (c) $p = 3$.

Figure 5: Plots of KNN Classification Rate as a function of $M \in \{10, 20, \ldots, 320\}$, for $K \in \{1, 2, 3, 4, 5, 10\}$ and $p \in \{1, 2, 3\}$.

It remains to consider the classification rate as a function of the dimensionality, M, and the order of the Minkowski distance metric, p. Our conclusions regarding these two quantities are altogether more tentative. In particular, we note that what appears to be the maximum observed classification rate (0.9625) across all permutations of the input parameters, is achieved by different distance metrics, for different dimensions. The particular parameter combinations are summarised below

p	1	2	3
M	10	$50, 60, \ldots, 100$	$10, 190, 200, \ldots, 320$

TABLE 2: Input parameters for which the maximal observed classification rate, 0.9625, is achieved.

In light of these results, we can perhaps only reasonably conclude that, if we are solely interested in the performance of our classifier as determined by classification rate, and solely interested in the given data-set, our algorithm is not uniquely optimised by any single choice of input parameters,[6] but will perform most favourably for $K = 1$, and any permutation of parameters as specified in Table 2. This being said, it is perhaps notable that, specifying the Euclidean distance metric, the dimensions over which our classifier best performs precisely correspond with those recommended in

[5] In fact, the choice $K = 2$ gives identical results. This is a direct result of the choice of method for 'breaking ties' (see Footnote 8).

[6] However, it should be noted that we have only run the algorithm for a discrete and finite set of parameter values: if we were to consider all dimensions M, one may indeed be determined as optimal.

the Section 3. Thus, if we are not purely interested in classification performance, we would perhaps tend towards the use of $p = 2$, and $B \in [50, 100]$.

4.2.2 General Recommendations

It is, in fact, of interest to make more general recommendations regarding how best to set up our algorithm for application in the context of face recognition, and to ensure that our recommendations are relevant for an independent set of similar data. For this purpose, we make use of k-fold cross validation, with $k = 10$.[7] In the interest of brevity, and more importantly since it coincides with our previous recommendations regarding dimensionality, we will herein restrict our attention to $B \in \{45, 50, \dots 100\}$, and thus to $p = 2$. For completeness, we also run the cross-validation scheme for $K \in \{1, 2, 3, 4\}$:

On the basis of the output results (Figure 6), our recommendation for optimal set-up of the KNN algorithm would utilise the Euclidean distance metric, $K = 1$ nearest neighbours, and a PCA basis of dimension $M = 45$.

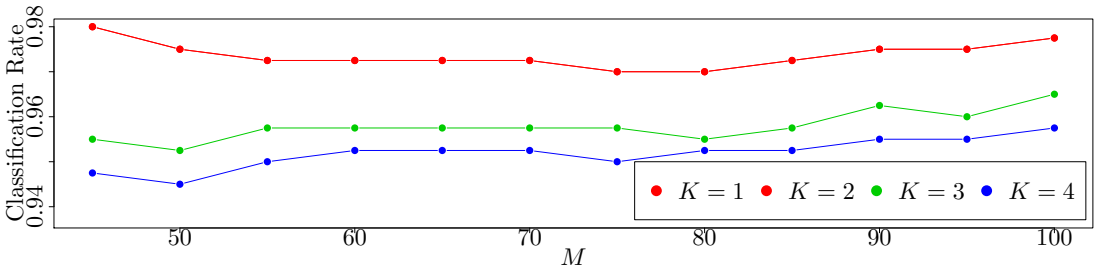


Figure 6: Plot of KNN Classification Rate as a function of $M \in \{45, 50, \dots, 100\}$, for $K \in \{1, 2, 3, 4\}$, with values computed using 10-fold cross validation.

[7] This choice of k is primarily motivated by computational considerations.