Pattern Recognition: Coursework 1

Georgios Solidakis: 00975189
Imperial College London
georgios.solidakis15@imperial.ac.uk

Thibaud Lemaire: 01618638
Imperial College London
thibaud.lemaire18@imperial.ac.uk

1. Introduction

The aim of this report is to investigate the performance of different algorithms in face recognition. Both supervised and unsupervised machine learning algorithms are applied in order to reduce the dimensionality of the data and distinguish between different people which constitute the labeled classes. By extracting relevant information from an unlabeled test image we can compare it to the information of labeled training images and assign it a class. Results on the relevant accuracy and time efficiency of each method are presented.

2. Dataset and Pre-Processing

The dataset contains a total of $N = 520$ images of faces, each of resolution 46×56 pixels given in vector form. The images are labeled according to the person presented. The total number of labels is $L = 52$ meaning there are 52 distinct people in the dataset. Each image is already centered and normalized, meaning we do not need to perform any transformations or augmentations before processing. In order to train the model, the data is randomly partitioned at an 80:20 ratio after evaluation and validation, while making sure there is the same proportion of images per label in the two sets (stratified split). This ensures the classifier is not biased towards some identities.

3. Recognition by Eigenfaces

Eigenfaces is a technique for dimensionality reduction which involves obtaining information regarding the variation of the data and then projecting it to a smaller subspace spanned by the most significant variations [1]. Since the number of data is $N = 520$, significantly smaller than the feature space which is $46 \times 56 = 2575$, we can project it to a much smaller subspace of dimensions $M \ll 2576$. This is done to reduce the complexity and redundancy of the problem by only keeping vital information for recognition and classification. The new feature space has a base of rank M composed by a set of eigenvectors called eigenfaces. These vectors, also called principal components, can be linearly combined to create any of the faces with an error depending on the number of components kept.

3.1. Naive PCA

3.1.1 Data processing

After partitioning the dataset according to Section 2 making sure each label is equally represented, the result is $N_{train} = 416$ and $N_{test} = 104$ representing the training and testing sets cardinalities respectively. We first normalize the training data by calculating the mean face of the training set $\bar{X} = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} X_i$ shown in Figure 1 and subtracting it from every image to get $\Phi_i = X_i - \bar{X}$ where Φ_i is the normalized i^{th} image in the training set. We define the matrix $A \in \mathbb{R}^{D \times N_{train}}$ as the matrix whose columns are the aforementioned normalized training images. We then calculate the Covariance matrix as $S_n = \frac{1}{N} \sum_{i=1}^{N} \Phi_i \Phi_i^T = \frac{1}{N} A A^T \in \mathbb{R}^{D \times D}$ where $N = N_{train}$ and D remains 2576.

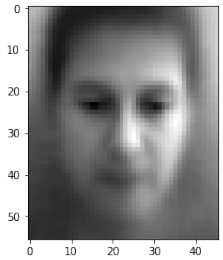

Figure 1. Mean face image of the 416 training data faces

3.1.2 Results

The dimension M of the subspace is chosen by selecting M eigenvectors of the covariance matrix S_n corresponding to the M greatest eigenvalues $\lambda_{1:M}$. Since the covariance matrix is real and symmetric the eigenvalues are also real and can be sorted by decreasing order which corresponds to a decreasing order of significance of the eigenvectors. It is known from linear algebra that $rank(AA^T) = rank(A)$ and $rank(A) = min(D, N) = 416$. Since matrix A is normalized by subtracting the mean image, the rank is reduced

by 1 meaning there are at most 415 non-zero eigenvalues.

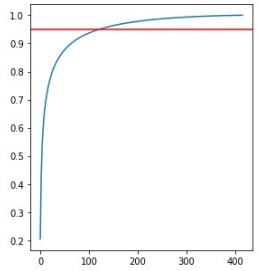

Figure 2. Cummulative sum of the eigenvalues of matrix S_n

Figure 2 shows the cumulative sum of the eigenvalues of S_n. Furthermore, we can deduce that 95% of the variance is explained by the first 121 eigenvalues. This means that the rest are much smaller and hence of much smaller importance allowing us to achieve great results by discarding any eigenvalue after that and setting $M = 121$.

Figure 3 shows the eigenvalues of the matrix S_n in descending order. It is visible from this plot that all the eigenvalues after the 415^{th} are almost zero. These eigenvalues are in fact zero but appear otherwise due to floating point errors.

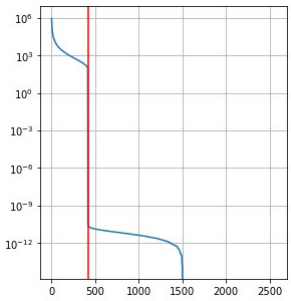

Figure 3. Eigenvalues of the matrix S_n in descending order

3.2. Efficient PCA

The covariance matrix used in the "Naive PCA" method in Section 3.1 is of dimensions $D \times D = 2576 \times 2576$. This is a very large matrix meaning that the computation of its eigenvectors and corresponding eigenvalues is quite expensive and inefficient, especially considering most of them are zero. This is due to the fact that eigendecomposition is a process whose execution time complexity is proportional to the cube of the dimension $\mathcal{O}(n^3)$ [4]. In order to avoid these computational costs, we can exploit the properties of symmetric matrices. It can be shown using linear algebra that for a symmetric matrix the pairs of eigenvalues and eigenvectors of $S_n = AA^T$ are related to those of $S_e = A^T A$. In this case, the matrix S_e is of dimensions $N \times N = 416 \times 416$, much smaller than S_n ($D \times D = 2576 \times 2576$) meaning that the execution time should dramatically decrease resulting in much greater computational efficiency.

3.2.1 Proof of equivalence

$$S_e \mathbf{v} = \lambda \mathbf{v} \Leftrightarrow \frac{1}{N} A^T A \mathbf{v} = \lambda \mathbf{v} \tag{1}$$

Multiplying both sides with A:

$$\frac{1}{N} A A^T A \mathbf{v} = \lambda A \mathbf{v} \Leftrightarrow S_n A \mathbf{v} = \lambda A \mathbf{v} \tag{2}$$

Renaming $\mathbf{u} = A\mathbf{v}$ we get:

$$S_n \mathbf{u} = \lambda \mathbf{u} \tag{3}$$

Therefore the matrices S_n and S_e have the same eigenvalues λ and their eigenvectors are related by the equation:

$$\mathbf{u} S_n = A \mathbf{v} S_e \tag{4}$$

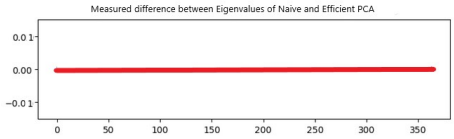

Figure 4. Measured difference between the eigenvalues of AA^T and $A^T A$

This is also confirmed experimentally by calculating the pairs for S_e. By summing the differences between the calculated eigenvalues of S_n and S_e we get 0, as shown in figure 4 above, meaning the eigenvalues of the two matrices are exactly the same. After applying the transformation (4) above by multiplying by A and normalizing we observe that the eigenfaces obtained are indeed the same too confirming the eigenvectors relation.

3.3. Comparison of Naive and Efficient PCA

In order to compare the two versions of PCA we attach a timer to each of them and measure the computation time as they run.The results are shown on Table 1 below.

Method	Execution time (s)
Naive PCA	22.72
Efficient PCA	0.23

Table 1. Execution times of the two PCA approaches

It is obvious from the results that the Efficient PCA is by far superior to the Naive PCA. Given that the eigendecomposition of both leads to the same pairs we will only use the efficient version throughout the rest of this paper.

4. Application of Eigenfaces

4.1. Reconstruction

In the previous section we used two versions of PCA in order to reduce the dimensions to M by choosing the M most significant eigenvector/eigenvalue pairs, and projected the image to the vector space spanned by it. In this section this process is inverted to reconstruct the image. Each face can be reconstructed as a linear combination of the M eigenvectors as shown below [2] .

$$\hat{X}_n = \bar{X} + \mathcal{U}\mathbf{w}_n \tag{5}$$

Where:

- $\hat{X}_n$ is the reconstructed n^{th} image
- $\bar{X}$ is the mean face of the training set
- $\mathcal{U}$ is the matrix containing the M eigenfaces as column vectors
- $\mathbf{w}_n$ is the vector of the n^{th} projection coefficients

Since some of the values have been discarded during PCA, losses have been introduced in the model. The reconstruction error, also called distortion measure, due to these losses can be calculated in the following way:

$$E = \frac{1}{N}\sum_{n=1}^{N}||X_n - \tilde{X}_n||^2 \tag{6}$$

This error can be minimized by selecting more and more eigenvalues. By choosing the M largest eigenvalues of S_e, the error E is proportional to the sum of the unused eigenvalues. $E \propto \sum_{n=\mathcal{M}+1}^{D}\lambda_n$ [3] . The actual proportionality relation is found to be equality experimentally by subtracting the error calculated with (6) from the sum of the remaining eigenvalues. The result of the aforementioned subtraction is a negligible value of order of magnitude 10^{-8} which is not zero due to floating point errors. The relationship between principal components and corresponding reconstruction error is shown below in Figure 5. It is obvious that the error increases exponentially by reducing the number of components. .

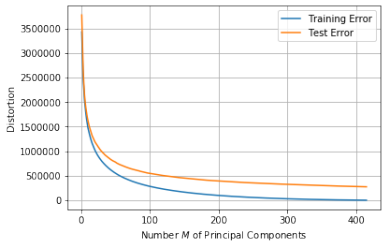

Figure 5. Reconstruction error of train and test sets for varying M

The training error approaches zero as we maximize the number of components used as expected considering the relationships mentioned above. On the other hand, the test error approaches a constant value as we maximize the number of components. This is due to the fact that the test image is projected on the eigenspace determined by the training data and only then it is reconstructed creating additional losses.

All of the above can be seen in Figures 14 and 15 in the Appendix which show the reconstruction of a random image for different values of M. By looking at the Figures, it is immediately noticeable that for a small M the reconstruction error is very large deeming it impossible to classify the reconstructed image. A huge improvement is witnessed when $M > 100$, with the reconstructed image seeming visually identical to the original. It is worth mentioning that the reconstructed test image is now visibly of lower quality than the training one due to the extra losses mentioned before. The test image appears to be blurrier but not to the point where classification is difficult.

4.2. Classification

Two methods of face recognition have been implemented. The Nearest Neighbour(NN) classification and classification using the reconstruction errors. These two methods are analyzed below.

4.2.1 Nearest Neighbour(NN) classification

Algorithm:

- Subtract the mean training image from the test image.
- Project normalized image to eigenspace.
- Compare the projected image to all the projected training images based on their distance.
- Select the closest training image and assign its label to the test image.

The NN algorithm is tested for various values of M and Neighbours in the search for the optimal values leading to the highest accuracy. The results are shown below in Figure 6. It is obvious from Figure 6 below that the highest accuracy occurs around $M = 60$ and is equal to **57.7%**. One can also witness over-fitting in this figure as the recognition accuracy on the test drops after we keep increasing M over a certain value. Figures 17, 18 and 19 in the Appendix show a success case along with a failure one and the confusion matrix obtained, making it clear that some of the labels (e.g 28) are more confusing (i.e more commonly predicted), and

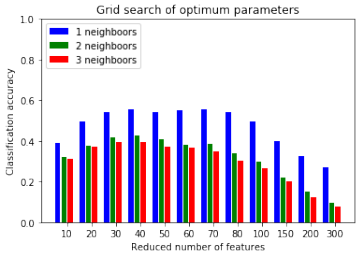

Figure 6. Variation of M, number of neighbours and recognition accuracy using NN classifier

4.2.2 Alternative method

The alternative method requires us to compute a new eigenspace for each class. This allows us to then project the test image in all of the eigenspaces computed above and calculate a set of reconstruction errors, one error per subset. We then proceed to classify the test image by assigning it the label corresponding to the eigenspace with minimum reconstruction error.

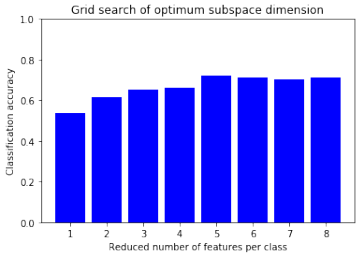

Figure 7. Grid search for optimal parameters varying the number of PCA features

Figure 7 illustrates a grid search for the optimal parameters. It plots the mean classification accuracy on the testing set depending on the number of PCA features we keep per class. It can be seen that the highest accuracy corresponds to 5 features per class and is equal to **71.1%**. This method outperforms the NN method significantly. Even though the accuracy achieved varies for different data pre-processing, it always outperforms the accuracy achieved by NN classification. The confusion matrix is shown in Figure 16 in the Appendix. By comparing it to the confusion matrix obtained by NN classification we can confirm the above.It is noticeable that there are many more yellow elements in the diagonal this time indicating a correct classification.

4.2.3 Comparison of methods

Even though we have already evaluated the two methods in terms of accuracy, we also need to evaluate them in terms of complexity and efficiency. The superior results of the alternative methods can be explained from the higher complexity of the model. The time complexity of projecting the matrix now also depends on the number of classes C. The NN classification model only depended on the number of bases used M and the number of features D with a time complexity of the order $\mathcal{O}(MD)$. The alternative method requires us to project the test image onto C distinct eigenspaces hence further increasing the time complexity order to $\mathcal{O}(MDC)$. Furthermore, the need to reconstruct the image each time contributes even more to the complexity. This time inefficiency was experimentally calculated by attaching timers to each method. After many executions, a general rule of thumb, was for the alternative method to take around 4 times longer than the NN method to execute. An example measured is presented below in Table 2 below.

Method	Execution time (s)
NN Classification	0.13
Alternative Classification	0.54

Table 2. Execution times of the two classification methods

5. PCA-LDA

So far we have successfully performed PCA to project the data to a lower dimension. PCA is an unsupervised algorithm with the goal of finding directions (Principal components) maximizing the variance in the dataset. To enhance these results and get a better accuracy, we further process the data with linear discriminant analysis (LDA). LDA, in contrast to PCA, is supervised and computes the directions (linear discriminants) that will represent the axes that maximize the separation between multiple classes. Overall, the following dimensionality reduction of the data D occurs: $D \longrightarrow M_{PCA} \longrightarrow M_{LDA}$.

5.1. LDA

LDA, besides reducing the dimension, maximizes the ratio of the between-class variance to that of the within-class variance. This leads to maximum separability of the data, making classification easier [5]. Algorithm:

- Calculate the between-class variance matrix S_B. S_B is the distance between the means of different classes and hence describes the separability between classes.
- Calculate the within-class variance matrix S_W. S_W is the distance between the mean and the samples of each class.
- Construct lower dimensional space maximizing S_B and minimizing S_W

S_B and S_W are defined as:

$$S_B = \sum_{i=1}^{c} n_i(\mu_i - \mu)(\mu_i - \mu)^T \tag{7}$$

$$S_W = \sum_{i=1}^{c} \sum_{j=1}^{n_i} (\mathbf{x}_{ij} - \bar{\mathbf{x}_i})(\mathbf{x}_{ij} - \bar{\mathbf{x}_i})^T \tag{8}$$

Where:

- μ_i represents the mean of the i^{th} class.
- μ represents the total mean of the data
- $\mathbf{x}_{ij}$ represents the i^{th} sample in the j^{th} class.
- c is the total number of classes

LDA is calculated using the Fisher Criterion:

$$S_W = \lambda S_B W \tag{9}$$

Where W is the transformation matrix and λ are its eigenvalues. The solution to this problem can only be calculated if S_w is non-singular. This is because the solution requires the inverse of S_W and is found by calculating the eigenvalues and eigenvectors of $W = S_W^{-1} S_B$ [5]. Each eigenvector calculated represents an axis of the LDA space and each associated eigenvalue its robustness. Similar to PCA the larger eigenvalues are more important as their magnitude reflects the ability to discriminate between different classes [5]. In order for the matrix S_W to be non-singular it needs to be of full rank.

5.2. LDA after PCA

Now that LDA has been defined, we can proceed from where we were left with PCA to further enhance our results. The training set has now been reduced to $X_{Train} \in \mathbb{R}^{N_{Train} \times M_{PCA}}$. Furthermore, the dimensions of the two scatter matrices are the same and equal to $M_{PCA} \times M_{PCA}$. Both matrices are symmetric and positive semidefinite. According to the previous section, the matrix S_W needs to be of full rank. When the dimensions are reduced by PCA we need to secure that this is true by upper-bounding it to $M_{PCA} \leq rank(S_W)$. It is expected from linear algebra that the upper bounds of the ranks of the two matrices are: $rank(S_W) \leq N_{Train} - c$ and $rank(S_B) \leq c - 1$. The rank of $W = S_W^{-1} S_B$ is equal to $min(S_W^{-1}, S_B)$. Given that in this case $c << N_{Train}$, it follows that $rank(W) \leq c - 1$. This means that there is an upper bound for M_{LDA} substituting for $c = 52$, that is $M_{LDA} \leq 51$. Furthermore, the upper bound of M_{PCA} can now be calculated as $M_{PCA} \leq rank(S_W) = N_{Train} - c = 364$. Subject to those two upper bounds, LDA can safely be performed after PCA and generate the so-called Fisherfaces. Each eigenvector of W corresponds to one Fisherface with a maximum of $c - 1$ Fisherfaces as calculated above.

5.3. Results

Applying PCA-LDA with NN classification using the boundaries calculated above and performing a grid search for the hyperparameters M_{LDA}, M_{PCA} resulted in Figure 8 below. It is visible from the square with the brightest yellow colour that the optimal parameters in this case are $M_{LDA} = 45, M_{PCA} = 150$. Using these results and testing the algorithm again we notice a further increase in recognition **accuracy to 88.5%**.

Figures 23 and 24 in the appendix illustrate a successful and a failed attempt in classifying respectively. Figure 22 in the appendix, illustrates the new confusion matrix confirming our result. The diagonal has more yellow points than ever before indicating a much higher accuracy. We can infer that the discriminatory information lies in the mean of the classes instead of the variance of the data as PCA suggested. The data has now shifted so that the data of each class are clustered closer to the mean of that class rather than the mean of the whole set or the means of other classes allowing better discrimination.

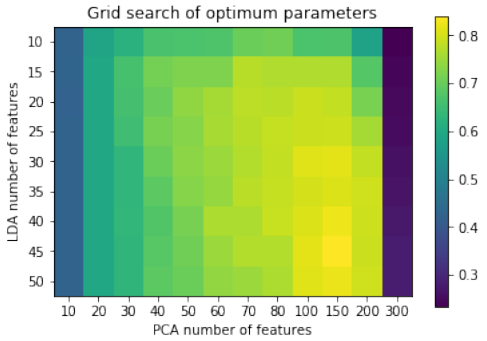

Figure 8. Grid search for optimal parameter M_{PCA}, M_{LDA}

6. PCA-LDA Ensemble

So far PCA and LDA have been applied to reduce our sets dimensions while trying to keep as much discriminatory information as possible. Even though our classification success rate has been increasing, these methods still suffer from problems. Firstly, the training sample set is small compared to our feature vector. Secondly, the performance has been subject to the dimension of the subspace [6]. In order to avoid overfitting and other problems we might encounter reducing the amount of discriminatory information, we now perform random sampling on feature vectors and training samples. As a result, multiple decorrelated classifiers are constructed and then combined together(fusion) to what is called a committee machine. In this way, nearly all the discriminatory information is preserved and we have a powerful decision rule leading to improved generalization and robustness [6].

6.1. General procedure

The idea is to train numerous weak classifiers on different datasets extracted from our training set and to choose a

rule to merge their estimations. In our case, each classifier is bootstraped from a base classifier and training data are drawn using random subsets of the original training samples and/or features. The base model is constructed by choosing the following: (1) Model specific parameters based on which of the two models below is applied, (2) The number of individual classifiers to be trained and finally (3) Randomization parameters such as the number of samples and the number of features used in training the model. Two different models are considered:
In the first model, kNN classification is used as the base model and its specific parameter is k.PCA-LDA is first performed on the whole training set and then the Fisherfaces obtained are used as the input of the Ensemble classifier. In this case, the randomization is performed on the M_{LDA} fisherfaces.
In the second one, PCA-LDA-NN is used as the base model to bootstrap. This means that the number of times PCA and LDA are performed is equal to the number of individual classifiers trained. The randomization is performed on the N_{Train} faces and their 46×56 features.
In both cases, we use the majority voting as fusion rule. Each classifier predicts(votes) a class and most frequent prediction is chosen.

6.1.1 Randomness

We can express randomness as $\rho = |\mathcal{T}|$ where $\mathcal{T}$ is the ensemble of all possible parameters of our model. Thus, using the binomial coefficient notation :

$$\rho = \binom{N_{samples}}{\lceil N_0 \times N_{samples} \rceil} \times \binom{N_{features}}{\lceil N_1 \times N_{features} \rceil} \quad (10)$$

Where N_0 is the proportion of samples we draw our training models from (max_samples in SkLearn) and N_1 is the proportion of features we draw our models from (max_features in SkLearn).

6.2. First base model : kNN only

As mentioned before, in this approach, PCA and LDA are first performed on the whole training set and the random patching (bagging and random subspace combination) is applied on the LDA features. $M_{PCA} = 150$ and $M_{LDA} = 50$ were chosen to minimize the loss of information based on previous results.
Afterwards, a gridsearch for optimal hyper parameters T, N_0 and N_1 is performed with cross validation. The results are shown in Figures 9 and 10 below. The gridsearch shows that the best classifier with this architecture is obtained using $T = 200$ estimators, all the training samples ($N_0 = 1.0$) and 60% of the 50 features ($N_1 = 0.6$). Using these results, **90.4% accuracy** is attained as shown in the confusion matrix of Figure 25 in the Appendix.

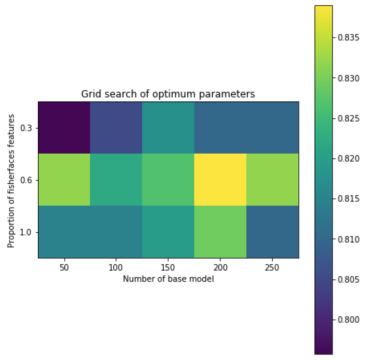

Figure 9. Grid search for optimal feature randomization and number of classifiers with $N_0 = 1.0$

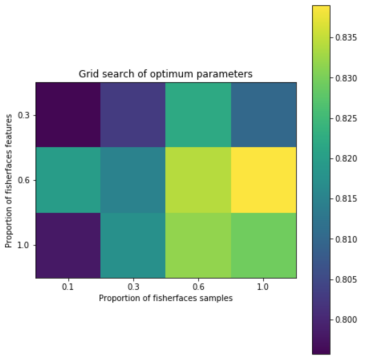

Figure 10. Grid search for optimal sample randomization and feature randomization using $T = 200$ base models

Using this model, the randomness parameter is not too high because $N_{features} = 50$. With the optimal parameters:

$$\rho = \binom{416}{1.0 \times 416} \times \binom{50}{0.6 \times 50} \approx 10^{13} \quad (11)$$

We can compare the average accuracy of base classifiers to the accuracy of the committee machine. With our optimal hyper parameters, mean base model accuracy is 74% giving an average error of $E_{av} = 26\%$ while the committee machine accuracy is 90% imputing an error of $E_{com} = 10\%$.

6.3. Second base model : PCA-LDA+NN

This time the base model is PCA-LDA+NN, that is we perform randomization on the raw dataset, perform PCA and LDA and then train our NN classifier as many times as there are estimators. The committee machine is created by combining once again all the individual models using majority vote.

Considering that the input data are much more complex than with the previous architecture, weak learners are used (i.e. based on a small subset). A gridsearch is performed again seeking for optimal values for T, N_0 and N_1. Large values of M_{PCA} and M_{LDA} can be used (regarding the feature subset) since the committee machine will take care of overfitting.

The gridsearch shown in Figures 11 and 12 below shows that the best classifier with this architecture is obtained using $T = 350$ estimators, all the training samples ($N_0 = 1.0$) and 4% of the 2576 features ($N_1 = 0.04$). Using these results, **98.1% accuracy** is attained as shown in the confusion matrix of Figure 26 in the Appendix.

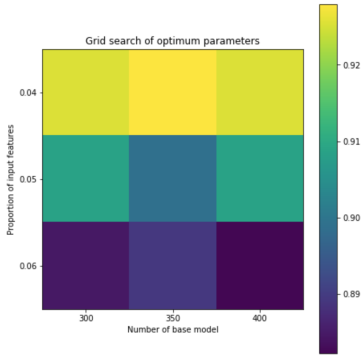

Figure 11. Grid search for optimal feature randomization and number of classifiers with $N_0 = 1.0$

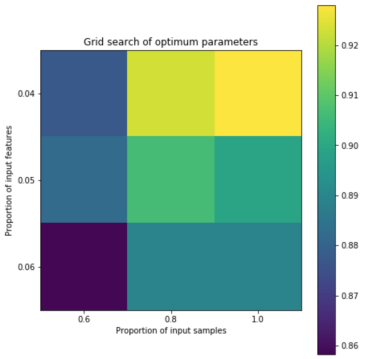

Figure 12. Grid search for optimal sample randomization and feature randomization using $T = 350$ base models

Using the model described, the randomness parameter is very large because we consider $N_{features} = 2576$. The numeric value of ρ is expected to be much higher than that of the previous architecture. This is confirmed by the numerical calculation below:

$$\rho = \binom{416}{0.8 \times 416} \times \binom{2576}{0.04 \times 2576} \approx 10^{277} \qquad (12)$$

Using this architecture, the base model mean error E_{av} is found to be $E_{av} = 35\%$ while the error of the committee machine is $E_{com} = 2\%$.

7. Conclusion

This report has investigated different methods of dimensionality reduction and classification, and their overall performance in face recognition. Two different approaches to dimensionality reduction are taken, PCA and LDA.

After evaluating PCA alone , which achieved moderate results, an alternative method using reconstruction errors is suggested significantly increasing the performance achieved. This increase in accuracy is owed to the greater complexity of that model. Eventually, PCA and LDA are combined reducing the dimensions sequentially while maximizing the information kept relative to discrimination. Using PCA for dimensionality reduction and then LDA for maximizing data separability, the combination of the two resulted in a much better accuracy.

Finally two distinct methods of ensemble learning are used to further enhance the results. This enhancement takes care of overfitting issues leading to a more stable and robust model. The first model sampling LDA features only increased accuracy by a small amount. On the other hand, the second model of ensemble learning using sampling of raw data significantly outperformed all of the above methods by achieving a 98.1%. The results of all the aforementioned methods are summarized in Table 3 below.

Method	Accuracy
PCA with kNN Classification	57.7%
Alternative method	71.1%
LDA after PCA with NN	88.5%
Ensemble - kNN only	90.4%
Ensemble - PCA-LDA+ kNN	98.1%

Table 3. Comparison of different methods and their respective classification accuracy

Appendix

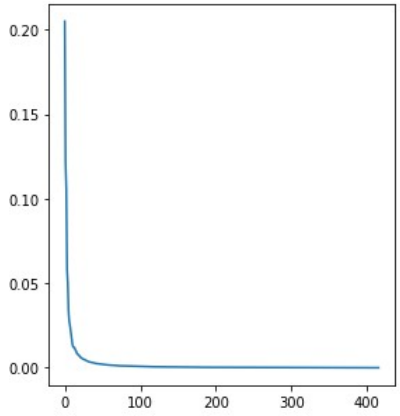

Figure 13. Eigenvalues of the covariance matrix sorted in decreasing order of magnitude

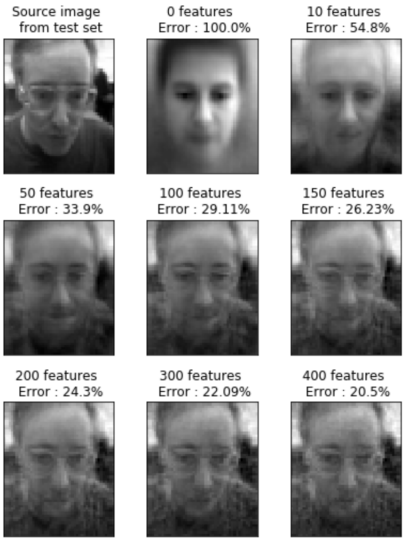

Figure 14. Reconstruction error of a **test image** for varying M_{PCA}

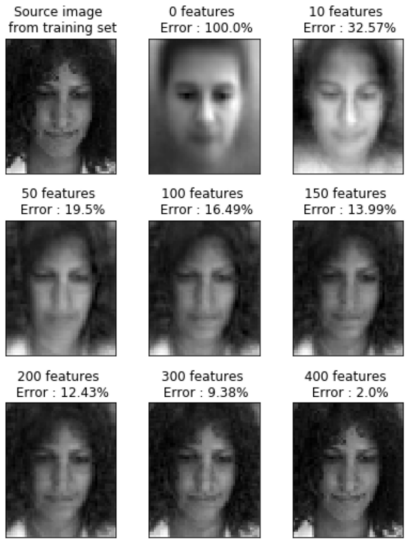

Figure 15. Reconstruction error of a **train image** for varying M_{PCA}

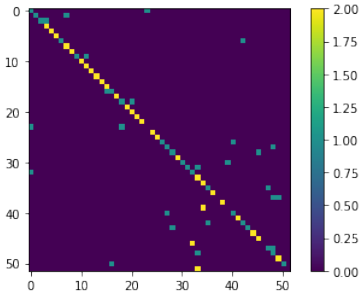

Figure 16. Confusion matrix of predicted and true labels using PCA+NN classification

Figure 17. Successful attempt using PCA+NN
Left - Test image, Middle - Reconstructed image in PCA space, Right - Predicted label

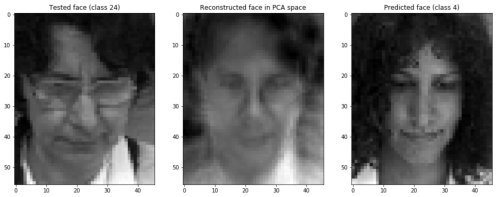

Figure 18. Failed attempt using PCA+NN
Left - Test image, Middle - Reconstructed image in PCA space, Right - Predicted label

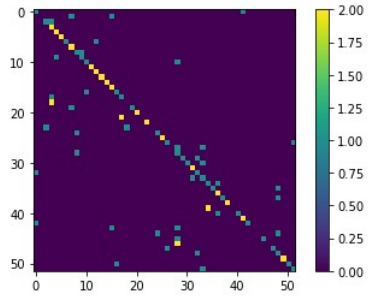

Figure 19. Confusion matrix of predicted and true labels using the alternative method

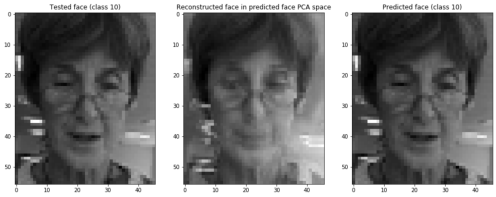

Figure 20. Successful attempt using the alternative method
Left - Test image, Middle - Reconstructed image in PCA class-subspace, Right - Predicted label

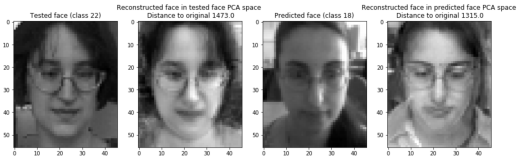

Figure 21. Failed attempt using the alternative method
From left to right: Test image, Reconstructed image in true class-subspace, Predicted label, Reconstructed image in predicted class-subspace

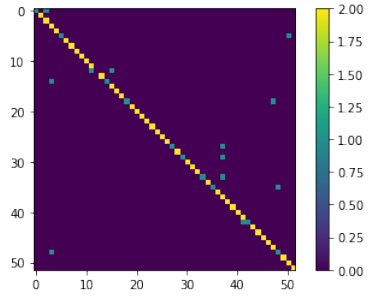

Figure 22. Confusion matrix of predicted and true labels using PCA-LDA+NN classification

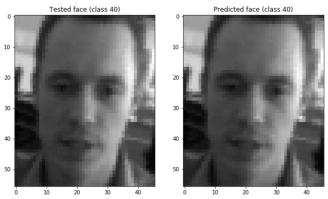

Figure 23. Successful attempt using PCA-LDA+NN
Left - Test face image, Right - Predicted face image

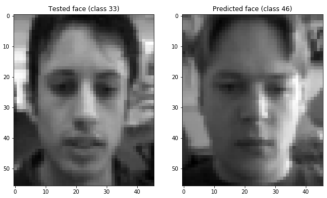

Figure 24. Failed attempt using PCA-LDA+NN
Left - Test face image, Right - Predicted face image

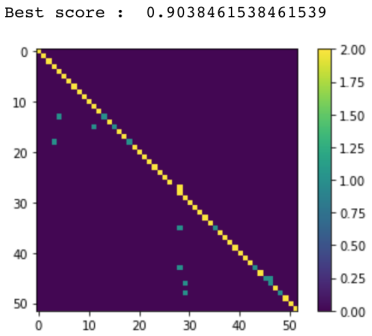

Figure 25. Confusion matrix of our Ensemble classifier using NN as base model

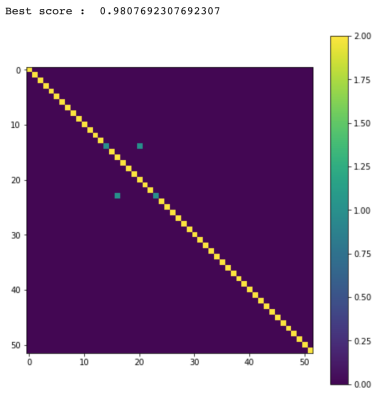

Figure 26. Confusion matrix of our Ensemble classifier using PCA-LDA+NN as base model

References

[1] Eigenface https://en.wikipedia.org/wiki/Eigenface

[2] Kim Tae-Kyun. *Face Recognition by Eigenfaces*. Accessed: 17-12- 2017.

[3] Kim Tae-Kyun. *Optimisation in pattern recognition: Principal Component Analysis (PCA)*. Accessed: 17-12-2017.

[4] Eigenvalue algorithm https://en.wikipedia.org/wiki/Eigenvalue$_a lgorithm$.

[5] Tharwat, Alaa Gaber, Tarek Ibrahim, Abdelhameed Hassanien, Aboul Ella. *Linear discriminant analysis: A detailed tutorial*. Ai Communications. 30. 169-190,. 10.3233/AIC-170729.

(2006). Random Sampling for Subspace Face Recognition. International

[6] Wang, Xiaogang Tang, Xiaoou. *Random Sampling for Subspace Face Recognition*. Journal of Computer Vision. 70. 91-104. 10.1007/s11263-006-8098-z.