Pattern Recognition Coursework 1

Yao Lei Xu[1] (01062231) and Alejandro Gilson[2] (01112712)

I. EIGENFACES

A. *Data Pre-Processing and Eigen-Decomposition*

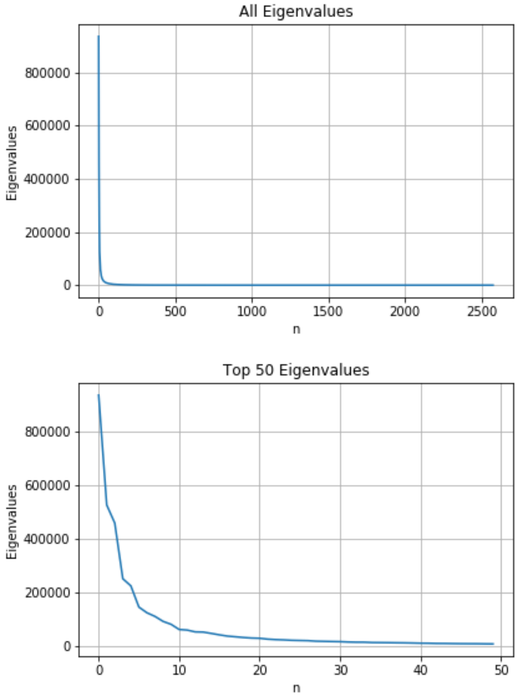

Fig. 1. Eigenvalues

The given data-set $\mathbf{x}$ is a 2-dimensional DxN matrix, where $D = 2576$ and $N = 520$. D is the dimension of each face vector which contains information about the pixels of a 46x56 image (46x56=2576 pixels), and there are $N = 520$ face samples in total. However, in order to treat the face recognition as a two dimensional problem, $\mathbf{x}$ is organized such that there are 10 different pictures for the same person that have slight differences in orientation. This allows for the training to be faster and simpler than if it were trying to recover the 3 dimensional properties of the faces.

To partition $\mathbf{x}$ into training and test sets without any orientation bias for the learning, 8 out of 10 pictures per person are randomly chosen to be in the training set, while the remaining 2 in the test set. The selection of an appropriate training data ratio is of vital importance for the performance of a face recognition algorithm. The training ratio should be high to increase the accuracy of the algorithm but low enough to have a good estimate of the accuracy on the test data. This is, therefore, a trade-off of accuracy and our ability to measure it. For this whole coursework, a training ratio of 0.8 will be used with 104 and 416 test and training faces respectively (from here onwards, N=416 unless otherwise stated). For each image calculated wrong the accuracy will be 0.96% lower. This makes it an appropriate estimation of the real accuracy of the algorithm.

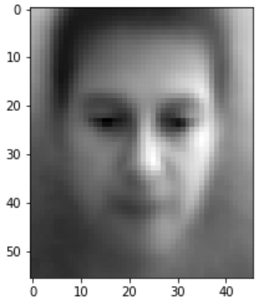

Fig. 2. Average Training Set Face

The normalized faces matrix $\mathbf{A}$ is obtained by subtracting the average value from the training set (the average face in shown in figure 2). From there, the eigenvalues and the eigenvectors are obtained from the covariance matrix $\mathbf{S} = \frac{1}{N}\mathbf{A}\mathbf{A}^T$, where $\mathbf{S}$ is a DxD matrix. Mathematically speaking, the matrix $\mathbf{S}$ captures the differences among faces and their global mean, and its eigenvectors project to a vectors space that can capture these features. As shown in Figure 1, the ordered eigenvalues corresponding to different eigenvectors decrease rapidly in magnitude, which means that there are some features for which the variance among facial features is higher than the rest. This allows us to distinguish different faces.

In summary, the eigenvectors are normalized vectors that map the feature space, while the eigenvalues represents the amount of variation in that feature space. Therefore, the physical meaning behind this is that there are only a few facial features (eigenvectors) that contain meaningful (non zero) information. Therefore, for recognition purposes, the first few eigenvectors can be used for satisfying performance, since the smaller eigenvalues are near zero in value (not exactly zero due to floating point precision). For reconstruction however, the number of eigenvectors used should be as high as possible to capture all the details.

B. *Low-Dimensional Computation*

For the low-dimensional computation, the covariance matrix is computed as:

$$\mathbf{S}_2 = \frac{1}{N}\mathbf{A}^T\mathbf{A}, \quad where \mathbf{S}_2\ is\ a\ N \times N matrix \tag{1}$$

In this case, we can obtain only N eigenvectors given the size of the covariance matrix. However, the top N eigenvectors $\mathbf{u}_i$ of $\mathbf{S}$ are related to the eigenvectors $\mathbf{v}_i$ of $\mathbf{S}_2$ as $\mathbf{u}_i = A\mathbf{v}_i$ (after normalizing the magnitude to one), while their eigenvalues are the same (the average absolute differences between them are 3.76e-11). For practical datasets where $D >> N$, the low-dimensional computation has the advantage of being more computationally efficient as the computational time grows with bigger matrices with complexity $O(n^3)$ assuming naive matrix multiplication algorithm (the time taken to compute S and S_2 are 7.25s and 0.10s respectively). However, less eigenvectors (facial features) tend to have worse performance due to loss of information for both face reconstruction and face recognition (see section III), but the error is negligibly small as shown by Fig.1.

II. APPLICATION OF EIGENFACES

A. Face Image Reconstruction

The face reconstruction is carried out by the weighted sum of the facial features and the average face:

$$\mathbf{f}_{n\_rec} = \mathbf{f}_{avg} + \sum_{i=1}^{M} a_{ni}\mathbf{u}_i, \tag{2}$$

where $a_{ni} = \phi_n^T \mathbf{u}_i$ *and* ϕ_n *is the normalized training face*

For this reason, the more facial features (eigenvectors) there are, the better the reconstruction will be since more information can be captured. Figure 3 shows the original and reconstructed faces across different M values. For $M = 10$, the reconstructed face resembles the training set average with little added features, while for $M = 400$ the reconstructed face is qualitatively the same as the original. Figure 4 shows the pixel-wise mean absolute error between the original face and the reconstructed face (averaged across all training set examples) as a function of M. The drastic decrease in performance with increasing M is expected sine the face features decrease drastically in terms of eigenvalue as well.

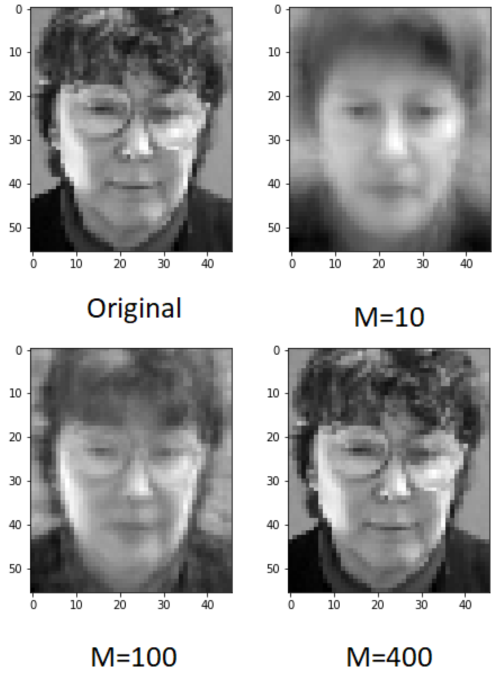

Fig. 3. Face Image Reconstructions

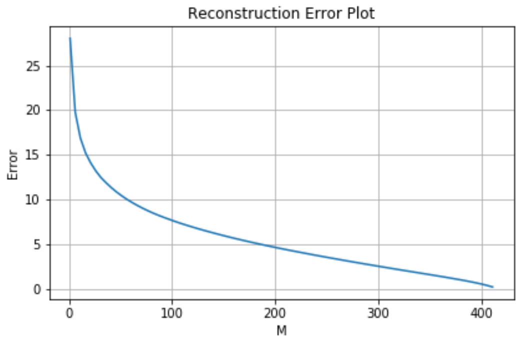

Fig. 4. Reconstruction Error

B. Face recognition

The decomposition of faces into subspaces constructed by their principal eigenvectors is of great importance in the area of face recognition. Following this method, the most important features of each face are extracted and this information can be used to investigate the identity of an individual. In order to do so, two approaches will be studied: Nearest Neighbour classification and an alternative method. To prepare the data for face recognition, all faces in the data-set are assigned labels so that when a match is found we can identify which person the face image belongs to. In both methods, a low dimensional approach will be followed.

1) Nearest Neighbour classification: When a face is projected into the principal eigensubspace of the training set of faces, the projection tends to lie close to other faces whose features resemble that of their own. In most of the cases, those other faces belong to the same person. However, sometimes two different individuals have similar features that make the algorithm guess wrong. An example of failure case can be seen in figure 5. In order to perform NN classification, we first select a test face whose identity we want to obtain and normalize it by subtracting the training set mean. As before, we obtain the face subspace of the whole training set and select the M largest eigenvectors. Each training face is projected into the subspace by multiplying the normalized face with the selected eigenvectors. The classification error is then defined as in equation 3. The identity of the training face that minimizes the error is said to be the identity recognized by the algorithm.

$$e = min_n ||\omega - \omega_n||, \quad n = 1, \ldots, N \tag{3}$$

As it can be observed in figure 6, the algorithm has a higher accuracy with larger values of M until it reaches approximately 40. After this point any larger values have almost no effect on the accuracy. The maximum accuracy

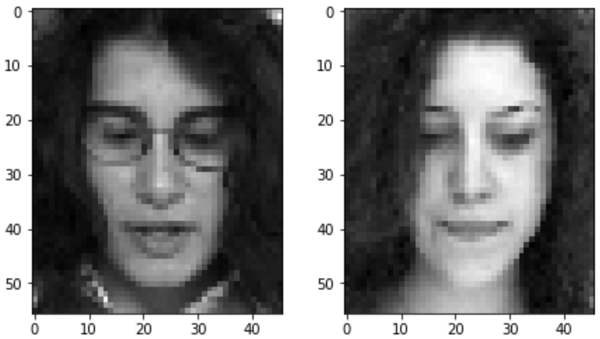

Fig. 5. Left: Face to be identified. Right: Wrongly identified individual

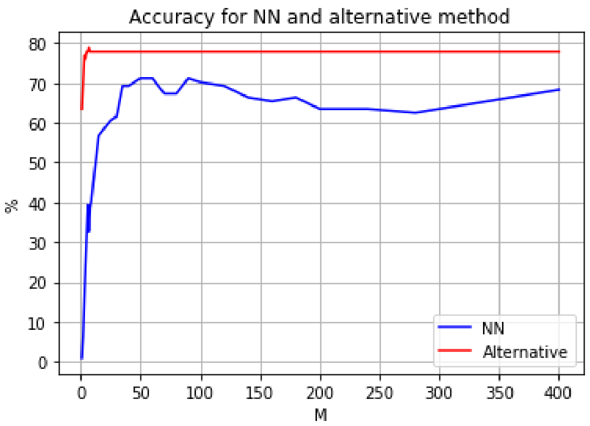

Fig. 6. Face recognition accuracy for NN and alternative method for different values of M

is 72% for $M = 50$. The steady state value can be observed to be approximately 65%. This leaves some room for improvement. The results for the computation time are displayed on figure 7. It can be seen that NN classification is faster and yields lower accuracy than the alternative method for the same values of M. However, as M increases, so does the computation time of the NN algorithm because more eigenvectors and are used to compute the eigenspace.

2) Alternative method for classification: In this method the subspace is calculated per class, i.e. identity. In order to do so, the average face of all the training faces belonging to the same person is taken and its eigenspace is calculated using equation 1. The test image that is to be recognized is projected into each of the sub-spaces and the one whose reconstruction error is minimum is taken to be the recognized identity. N eigenvectors and eigenvalues describe the sub-space of each identity, therefore, for any $M \geq N$ the results will be the same. This can be observed in figure 6 for all values $M \geq 8$. This method outperforms the NN algorithm in accuracy for any value of M, which is expected as the PCA learning takes into account class specific information. The maxium accuracy is 78% for $M = 7$. Confusion matrices for both the NN and the alternative method are displayed in figure 8 and 9.

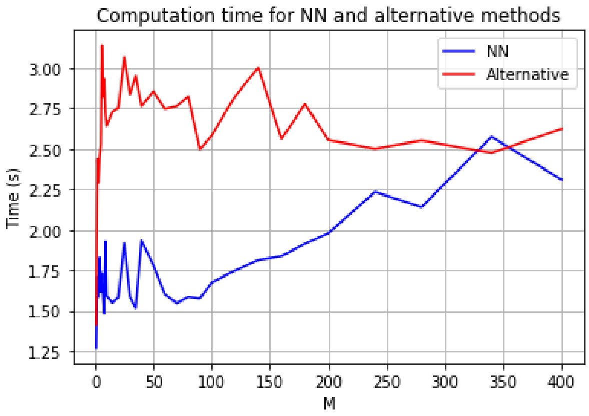

Fig. 7. Computation time for NN and alternative method for different values of M

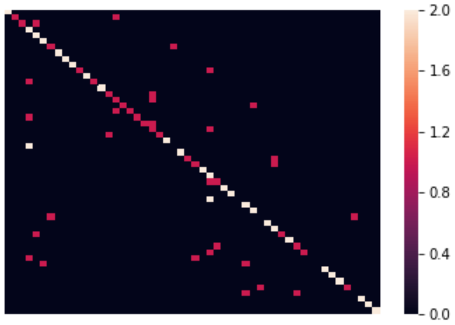

Fig. 8. Confusion matrix of the NN algorithm for M=150

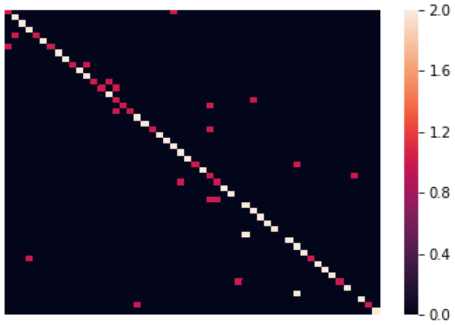

Fig. 9. Confusion matrix of the Alternative algorithm (M=8)

III. THE PCA-LDA OPTIMIZATION PROBLEM

A. *Drawbacks of PCA-LDA Classification*

The idea behind PCA-LDA method is to extract the face features with greatest variation from the input faces and then use those features to discriminate among different classes of people. The optimization problem for the PCA and LDA methods are respectively:

$$\mathbf{W}_{pca} = argmax_{\mathbf{W}}|\mathbf{W}^T\mathbf{S}_T\mathbf{W}| = argmax_{\mathbf{W}}|F_{PCA}(\mathbf{W})| \tag{4}$$

$$\mathbf{W}_{lda} = argmax_{\mathbf{W}}\frac{|\mathbf{W}^T\mathbf{S}_B\mathbf{W}|}{|\mathbf{W}^T\mathbf{S}_W\mathbf{W}|} = argmax_{\mathbf{W}}|F_{LDA}(\mathbf{W})| \tag{5}$$

Where the total scatter matrix $\mathbf{S}_T$, within class scatter matrix $\mathbf{S}_W$ and between class scatter matrix $\mathbf{S}_B$ are defined as:

$$\mathbf{S}_T = \sum_{n=1}^{N}(\mathbf{x}_n - \mathbf{m})(\mathbf{x}_n - \mathbf{m})^T$$

$$\mathbf{S}_W = \sum_{i=1}^{C}\sum_{x\in C_i}(\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T$$

$$\mathbf{S}_B = \sum_{i=1}^{C}(\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T$$

With global mean $\mathbf{m}$ and *i-th* class mean $\mathbf{m}_i$.

Given that $\mathbf{W}^T_{opt} = \mathbf{W}^T_{lda}\mathbf{W}^T_{pca}$, by substitution, the PCA-LDA classification yields the optimization problem:

$$\mathbf{W}_{lda} = argmax_{\mathbf{W}}\frac{|\mathbf{W}^T\mathbf{W}^T_{pca}\mathbf{S}_B\mathbf{W}_{pca}\mathbf{W}|}{|\mathbf{W}^T\mathbf{W}^T_{pca}\mathbf{S}_W\mathbf{W}_{pca}\mathbf{W}|} \tag{6}$$

However, the major drawback of equation 6 is that it performs the PCA and LDA serially, which means that the PCA projection might have already lost some important discriminative information before LDA is applied.

B. *Combined PCA-LDA Optimization*

To overcome the problem in the sequential PCA-LDA method, we can combine equations 4 and 5 to strike a balance between the two method simultaneously by forming the following optimization problem:

$$\mathbf{W}_{opt} = argmax_{\mathbf{W}}(|F_{opt}(\mathbf{W})|)$$

$$\mathbf{W}_{opt} = argmax_{\mathbf{W}}(\alpha|F_{LDA}(\mathbf{W})| + \beta|F_{PCA}(\mathbf{W})|) \tag{7}$$

Where α and β are proportional to how much importance we want to give to the discriminative and the generative parts respectively. Then, by setting $\mathbf{W}^T\mathbf{W} = 1$ as constrain like in the PCA case, we can formulate the Lagrangian as:

$$L = F_{opt}(\mathbf{W}) + \lambda(1 - ||\mathbf{W}||)$$

$$L = \alpha\frac{\mathbf{W}^T\mathbf{S}_B\mathbf{W}}{\mathbf{W}^T\mathbf{S}_W\mathbf{W}} + \beta\mathbf{W}^T\mathbf{S}_T\mathbf{W} + \lambda(1 - \mathbf{W}^T\mathbf{W}) \tag{8}$$

By setting the derivative of L w.r.t. $\mathbf{W}$ to zero, we obtain:

$$\nabla_{\mathbf{W}}F_{opt}(\mathbf{W}) = \lambda\mathbf{W} \tag{9}$$

Hence, to find the projection with balanced generative and discriminative aspects, we need to find $\mathbf{W}$ such that the equation 9 holds.

IV. LDA ENSEMBLE FOR FACE RECOGNITION

Linear Discriminant Analysis (LDA) is a rationalization of the Fisher's Linear Discriminant (FLD) algorithm. Its aim is to find the optimal direction that best separates data from different classes. This is equivalent to maximizing the numerator of equation 10 while keeping the denominator constant.

$$J(\mathbf{W}) = \frac{\mathbf{W}^T\mathbf{S}_B\mathbf{W}}{\mathbf{W}^T\mathbf{S}_W\mathbf{W}} \tag{10}$$

Where $\mathbf{S}_B$ and $\mathbf{S}_W$ are the between class and within class scatter matrices, respectively. The solution of this optimization problem are the eigenvectors of the matrix $\mathbf{S}_W^{-1}\mathbf{S}_B$ as shown in equation 11.

$$\mathbf{S}_W^{-1}\mathbf{S}_B\mathbf{W} = \lambda\mathbf{W} \tag{11}$$

A. *PCA-LDA*

The rank of the matrix $\mathbf{S}_B \in \mathbb{R}^{\mathrm{D}\times\mathrm{D}}$ and $\mathbf{S}_W \in \mathbb{R}^{\mathrm{D}\times\mathrm{D}}$ is at most $N - c$ and $c - 1$ respectively, which are the numbers of non zero eigenvalues. However, $N < D$ which means that the within-class scatter matrix often cannot be inverted for equation 11. For this reason, PCA is implemented to reduce the dimension of the matrix using equation 1 and taking the M_{pca} eigenvectors with largest eigenvalues. The equation 11 therefore becomes:

$$(\mathbf{W}^T_{pca}\mathbf{S}_W\mathbf{W}_{pca})^{-1}(\mathbf{W}^T_{pca}\mathbf{S}_B\mathbf{W}_{pca})\mathbf{w} = \lambda\mathbf{w} \tag{12}$$

Where the M_{lda} eigenvectors with the largest eigenvalues for the matrix $(\mathbf{W}^T_{pca}\mathbf{S}_W\mathbf{W}_{pca})^{-1}(\mathbf{W}^T_{pca}\mathbf{S}_B\mathbf{W}_{pca})$ are used to construct the matrix $\mathbf{W}$. However, parameters M_{lda} and M_{pca} are limited by the rank of the scatter matrices (51 and 364 for $N = 416$ and $c = 52$). Finally, we project the data as in equation 13:

$$\mathbf{y} = \mathbf{W}^T\mathbf{x}_n, \quad \textit{where } \mathbf{x}_n \textit{ are the PCA face projections} \tag{13}$$

The results are shown in figure 10. A maximum accuracy of 87% is obtained for $M_{pca} = 250$ and $M_{lda} = 51$. It can be observed that generally, the larger M_{lda}, the better the accuracy. However, the sample faces contain noisy information and trying to represent them with a large M_{pca} makes the model unstable. For this reason, as this parameter increases so does the accuracy until a certain point at which it overfits the data and the accuracy is reduced. Figure 11 shows the confusion matrix for this method.

B. *Bagging Ensemble*

The main advantage of ensemble models is that by having randomly different models, they are increasingly more decorrelated among them, which increases the generalization performance when combining results. There are several ways of combining results, such as averaging, maximum value and

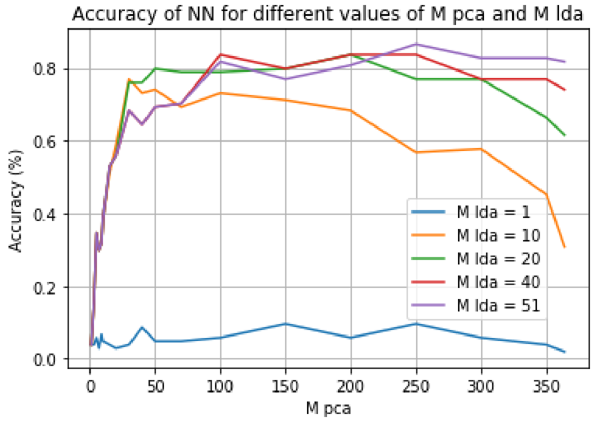

Fig. 10. Accuracy of NN LDA-PCA for different values of Mlda and Mpca

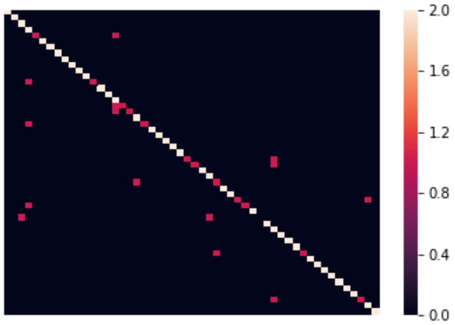

Fig. 11. PCA LDA Confusion Matrix $M_{lda} = 51$ $M_{pca} = 250$

minimum value. However, given that the class labels of the dataset indicate different people, the most sensible fusion rule is via majority voting.

Bagging (bootstrap aggregating) is a method of creating T different subsets of size s by sampling data from the original training set uniformly and with replacement (some samples repeat). For this exercise, we perform bagging on the classes such that for each of the T subsets of size $s = N$, there are about 63.2% of unique data. This allows the individual models to be trained and optimized for fewer samples. However, by combining their classifications by majority voting, the overall performance is expected to improve due to increased generalization. Figure 12 shows the accuracy for different T values for $M_{pca} = 250$ and $M_{lda} = 51$ (best performing parameters from the previous section), which increases until it reaches around 90% accuracy for $T > 100$. In contrast, the average individual classification performance is only about 37%.

C. Feature Space Randomization

Random sampling can also be computed on the feature space. Following this approach, a PCA projection of $N - 1$ eigenvectors is taken. From these eigenvectors, the $M0$ eigenvectors with largest eigenvalues are selected. From the remaining, $M1$ of them are chosen randomly (non-repeated). A matrix $\mathbf{W}_{pca}$ is constructed with these collected eigenvectors and LDA is computed in parallel with each of them for $M_{lda} = 51$ (this value was observed to give the best accuracy in the PCA-LDA algorithm). For face recognition, the NN algorithm is implemented for a T number of models and the recognition is a result of a majority vote. The accuracy is shown in figure 14.

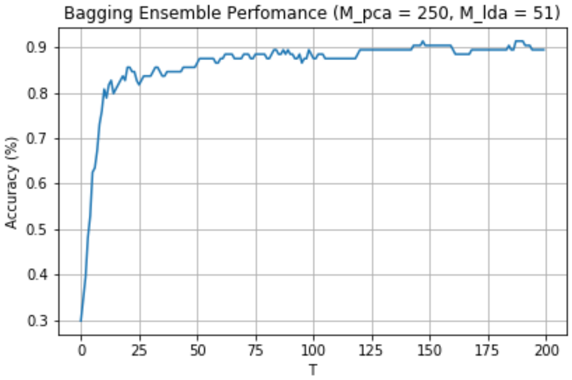

Fig. 12. Accuracy of Bagging Ensemble for increasing T

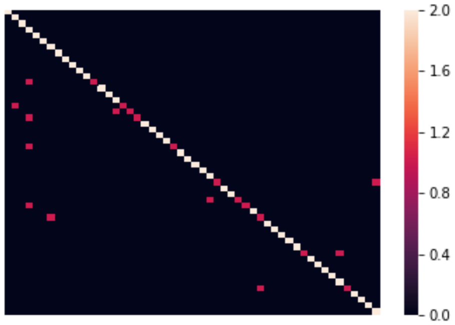

Fig. 13. Confusion Matrix of Bagging Ensemble (T=200)

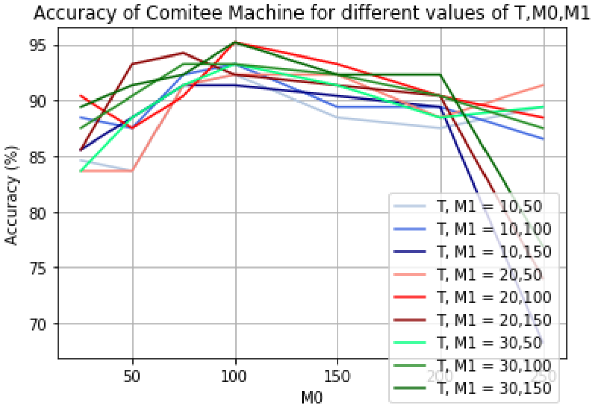

Fig. 14. Accuracy of Feature Space Randomization for different values of $M0, M1, T$

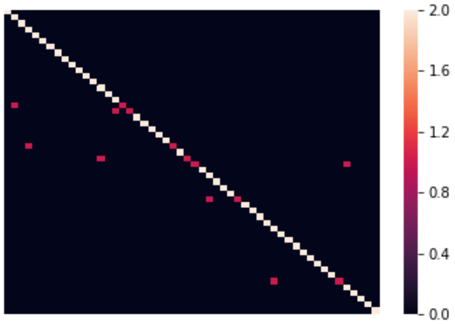

Fig. 15. Feature randomization confusion matrix

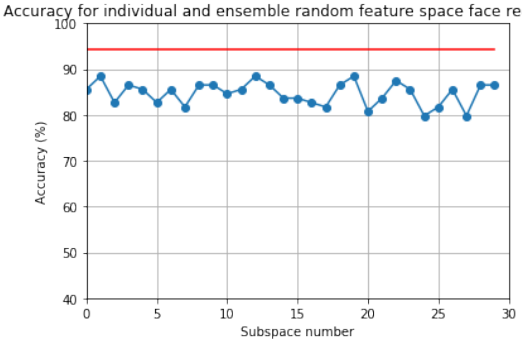

Fig. 16. Accuracy for individual and ensemble random feature space NN face recognition $M_{lda} = 51, M_{pca} = 415, T = 30, M0 = 100, M1 = 150$

The ideal committee machine has models that are uncorrelated. This attempted by selecting $M1$ random eigenvectors remaining eigenvectors $M_{pca} - M0$. Increasing the value of this parameter will yield big and somewhat independent subspaces. Since eigenvectors are sorted, if this parameter is too large, then the subspaces will become very big but correlated because there can only be $N - 1$ eigenvectors. On the other hand, if $M1$ is too small, the subspace will be smaller and so will the independent components of the subspace. The accuracy in both cases will be reduced. This phenomenons can be observed in the graph 14. For low and high $M1$, the accuracy is low because voting among similar subspaces provides no new perspectives. In this algorithm $M0$ also plays an important role. This parameter is kept relatively high to keep the main features of faces while allowing for $M1$ to have a relative importance. It can also be observed that the larger tbe number of models T the better the accuracy. However, other empirical results show that the accuracy stops increasing significantly for $T > 30$. For this dataset, a maximum accuracy of 95% is obtained for the parameters with values $M_{lda} = 51, M_{pca} = 415, T = 30, M0 = 100 \; and \; M1 = 150$. The comparison between individual and ensemble accuracy is shown in figure 16. It is clear that combining multiple models yields higher accuracy. Figure 15 shows the confusion matrix for the result.

V. Conclusions

In this experiment we studied several algorithms used for compressing data, reconstructing it and for face recognition. We started evaluating PCA for the acquisition of eigenfaces. The accuracy of all studied methods is shown in table I. The algorithm that outperforms all others is clearly the PCA-LDA implemented with a feature space randomization. Code is available on request.

Algorithm	Accuracy (%)
PCA NN	72
PCA Alternative	78
PCA-LDA	87
PCA-LDA bagging ensemble	90
PCA-LDA feature space randomization	95

TABLE I
Accuracy of all methods

References

[1] Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P.and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E.. Scikit-Learn Journal of Machine Learning Research

[2] Xiaogang Wang and Xiaoou Tang. Random Sampling for Subspace Face Recognition. Department of Information Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong. Received February 24, 2005; Revised November 27, 2005; Accepted January 4, 2006.