Decision Fusion for Patch-Based Face Recognition

Berkay Topçu and Hakan Erdogan
Faculty of Engineering and Natural Sciences,
Sabanci University, Orhanli Tuzla, 34956,
Istanbul, Turkey.
berkayt@sabanciuniv.edu, haerdogan@sabanciuniv.edu

***Abstract*—Patch-based face recognition is a recent method which uses the idea of analyzing face images locally, in order to reduce the effects of illumination changes and partial occlusions. Feature fusion and decision fusion are two distinct ways to make use of the extracted local features. Apart from the well-known decision fusion methods, a novel approach for calculating weights for the weighted sum rule is proposed in this paper. Improvements in recognition accuracies are shown and superiority of decision fusion over feature fusion is advocated. In the challenging AR database, we obtain significantly better results using decision fusion as compared to conventional methods and feature fusion methods by using validation accuracy weighting scheme and nearest-neighbor discriminant analysis dimension reduction method.**

***Keywords*-face recognition, patch-based face recognition, decision fusion, linear combiner training.**

I. Introduction

Face recognition is one of the most addressed pattern recognition problems in recent studies due to its importance in security application and human computer interfaces. Despite the intense research efforts on face recognition, it is still a difficult problem in real-world applications. Recognition of face images acquired in an outdoor environment with changes in illumination, partial occlusion and pose remains a largely unsolved problem [1]. To overcome these problems, patch-based face recognition was introduced [2].

In patch-based approaches, each image is divided into overlapping or non-overlapping regions called patches and local features are extracted from each region. One approach in patch-based face recognition is to concatenate features extracted from different patches in order to create the visual feature vector of a face image. In addition, features extracted from each patch can be classified separately and the recognition results are combined by decision fusion.

Patch-based face recognition and decision fusion in face recognition is a relatively new research topic. There are some previously proposed methods for patch-based face recognition. In study of [3], feature fusion (feature concatenation) and block selection with similarity measures are proposed. In [4], classification results of patches are weighted in which the weights are calculated from correct classification rates on probe set samples. In [5], similarity between any two faces are calculated over patches and final similarity is calculated by averaging the results of separate patches, which can also be replaced by a weighted average. Subspace methods are also employed on patch-based face recognition and as in [6], the classification results of patches and random subspaces are combined by majority voting. In [7], classifiers trained from separate patches are combined by a weighted summation as a first layer decision maker. In the second layer, decision of local ensemble classifiers is combined with global classifier trained from the whole face. However, the selection or calculation of the weights are not clear.

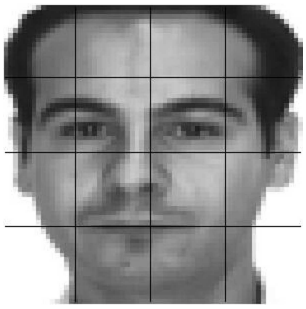

Figure 1. 16x16 blocks on a detected face from the AR database

In this study, we propose novel weighting schemes for combining classification results of classifiers which are trained over separate patches on face images. We name these three methods as Fisher weighting, support vector machine (SVM) weighting and validation accuracy weighting.

II. Patch-Based Face Recognition

Let a grayscale face image be a real-valued function $x : \Omega \to R$ defined on $\Omega = \{(i, j) : i \in [N_i], j \in [N_j]\}$, where $[N] = \{1, \ldots, N\}$. We consider a collection $\mathcal{S}$ of subsets of Ω given by

$$\mathcal{S} = \{S_k \subset \Omega : k \in [N_p]\}. \quad (1)$$

We define each S_k as a patch domain and take the face function restricted to kth patch domain as the face patch $x_k : S_k \to R$. Usually the patches would be non-overlapping and $\mathcal{S}$ would be a covering for Ω, but this is not a requirement and we may use overlapping or non-covering patches. Using non-overlapping rectangular regions (blocks) as patches is a common practice. We have conducted experiments on both overlapping and non-overlapping blocks and we have seen

that non-overlapping blocks provided higher recognition rates. Selection of block size is important because blocks should be big enough to provide sufficient information about the region it represents and should be small enough to provide stationarity and to prevent complexity in feature extraction. An example of blocks with block size of 16×16 is illustrated in Figure 1. We have conducted our experiments on both 8×8 and 16×16 block sizes and concluded that 16×16 block size is suitable for our approach.

A. Dimensionality Reduction and Normalization Methods

Decreasing the number of features of a multidimensional data under some constraints is desired in many applications. For dimension reduction, multidimensional data is projected or mapped into a space with less number of dimensions. Therefore, by applying a dimension reduction method, a d-dimensional data is mapped or transformed into a p-dimensional data, where $p < d$. In this study, we have used some well-known dimension reduction methods, discrete cosine transform (DCT) [8], principal component analysis (PCA) [9], and a recently proposed method nearest neighbor discriminant analysis (NNDA) [10].

When we use blocks in patch-based face recognition, every image is processed over non-overlapping square blocks. We define an image in a vector form as $\boldsymbol{x}^T = [\boldsymbol{x}_1^T ... \boldsymbol{x}_B^T]$ where B is the number of blocks and $\boldsymbol{x}_b$ denotes the vectorized b^{th} block of the image. For dimension reduction, we try to find a linear transform matrix for each block, $\boldsymbol{W}_b$, such that $\boldsymbol{f}_b = \boldsymbol{W}_b \boldsymbol{x}_b$. Then for each image, the feature vector is formed as $\boldsymbol{f}^T = [\boldsymbol{f}_1^T ... \boldsymbol{f}_B^T]$. On features extracted from separate blocks, we have applied some normalization methods that are described in [11].

B. Classification Method: Nearest Neighbor Classifier

In our face recognition experiments, we use nearest neighbor classification with one nearest neighbor. The choice of nearest neighbor classifier instead of other type of classifiers is due to the nature of the face recognition problem. Data obtained from face images are sparse therefore for other type of classifiers, extracting a statistical pattern that represents the nature of training data, is a difficult task.

In our experiments we have used nearest neighbor classifier with L_2-norm as the distance metric. Decision fusion requires extraction of class posterior probabilities $p(C_i|\boldsymbol{x})$ for the classifiers used. For nearest neighbor classifier, it is not immediately clear how to assign posterior probabilities. Following [12], we calculated the class posterior probabilities depending on the distance of $\boldsymbol{x}$ to the nearest training sample from each class. If we denote this distance vector as $\boldsymbol{D} = [\boldsymbol{D}(1), \boldsymbol{D}(2), ..., \boldsymbol{D}(N)]$, posterior probabilities associated with class i is calculated as:

$$p(C_i|\boldsymbol{x}) = \text{norm}(\text{sigm}(\log(\sum_{j \neq i} \boldsymbol{D}(j)/\boldsymbol{D}(i)))), \quad (2)$$

where $\text{sigm}(x) = (1 + \text{e}^{-x})^{-1}$. Class posterior probabilities are normalized to sum up to 1.

III. Decision Fusion

Decision fusion or classifier combination can be interpreted as making a decision by combining the outputs of different classifiers for a test image. In our case, instead of different type of classifiers, we combined outputs of nearest neighbor classifiers trained by different blocks that correspond to different regions on a face image.

For 16x16 blocks, we have 16 different block positions and a separate nearest neighbor classifier is trained by using the features extracted over the training data for that block. From a given test image, 16 feature vectors each corresponding to a different block are extracted. For each test image, local feature vector is given as an input to the corresponding classifier and the outputs of the classifiers are then combined to make an ultimate decision for the test image.

Unlike fixed combination methods, trainable combiners use the outputs of the classifier, class posterior probabilities, as a feature set. From the class posterior probabilities of several classifiers each corresponding to a block, a new classifier is trained to provide an ultimate decision by combining the posteriors To train a combiner, training dataset is divided into two parts as train and validation data. Individual classifiers are trained using the training data part. Then, the class posterior probabilities for each block are calculated on the validation data. For each image, these posterior probabilities are concatenated into a long vector ($[p(C_1|\boldsymbol{x}_1), p(C_2|\boldsymbol{x}_1), ..., p(C_{N-1}|\boldsymbol{x}_B), p(C_N|\boldsymbol{x}_B)]^T$) which is then used to train the combiner. However, the length of input feature vectors of the combiner, makes it difficult to train a classifier for multi-class classification problems. Therefore, we did not prefer to build a conventional trainable combiner for decision fusion.

In sum rule, the posterior probabilities for one class from each classifier are summed. Similar to the sum rule, one can also perform weighted summation of posterior probabilities. Intuitively, we would like to weight successful classifiers more. It is not immediately clear how to learn those weights. So, we developed methods to determine those weights in a weighted sum rule.

If we denote the contribution or weight of each block with w_b and for a given sample $\boldsymbol{x}$ posterior probability of i^{th} class for the b^{th} block as $p(C_i|\boldsymbol{x}_b)$, weighted sum of posterior probabilities for class i is given by:

$$p(C_i|\boldsymbol{x}) = \sum_{b=1}^{B} w_b p(C_i|\boldsymbol{x}_b). \quad (3)$$

Note that weighted sum rule can also be considered under the umbrella of trainable combiners since the weights can be learned from data as we show in the following.

We consider three different methods for learning the weights. We compare these methods with equal weights (EW) which corresponds to the sum rule when we use a fixed weight of $w_b = 1/B$. For the other methods that are described in the following parts, training dataset is partitioned into two as train and validation. Using train part, classifiers are trained and by using validation part as input, class posterior probabilities from first level classifiers are obtained to calculate block weights.

A. Fisher Weighting (FW)

The first weighting scheme, which we name as Fisher weighting, depends on the posterior probability distribution of true and false labels. In this method, for a single sample in the validation dataset, class posterior probabilities are calculated and posterior probability of the true class (let's say true class is i) at each block, $(p(C_i|\boldsymbol{x}_b))$, (16x1 vector) is labeled as positive score. For a sample $\boldsymbol{x}$ in the validation data, positive score vector is shown as:

$$\mathbf{PS} = \left[\; p(C_i|\boldsymbol{x}_1) \;\; p(C_i|\boldsymbol{x}_2) \;\; ... \;\; p(C_i|\boldsymbol{x}_B) \; \right].$$

Remaining posterior probabilities of false classes, where $j = 1 : N$ and $j \neq i$, $[p(C_j|\boldsymbol{x}_1), p(C_j|\boldsymbol{x}_2), ..., p(C_j|\boldsymbol{x}_B)]$ are labeled as negative score vectors. For each sample, this procedure is repeated and positive score and negative score matrices are combined in order to create two datasets which consist of class posterior probabilities of blocks.

Our aim is to find a weight for each block so that successful blocks are weighted more. Fisher's discriminant (or linear discriminant analysis (LDA)) finds the linear combination of vectors, such that these vectors are most separated in the projected space. If we successfully project our positive score and negative score vectors to 1-dimension where they can be separated, we can use the coefficients used for this mapping as our weights for each block.

By combining these two datasets, we get a 16-dimensional two-class dataset. Then the dimension of this dataset is reduced to one from 16 by using LDA and elements of the resulting dimension reduction vector of LDA are used as block weights. Distribution of positive scores and negative scores, after projecting to 1-dimension is presented in Figure 2. Note that, this procedure may yield negative weights for some blocks which may be counter-intuitive. In practice, we observed some small negative weights in the weight vector, but this did not cause any problems.

B. SVM Weighting (SVM-W)

This weighting scheme has the same motivation as Fisher weighting, however, instead of employing LDA on score vectors, a linear support vector machine (SVM) is used for classifying positive and negative scores. This also yields a set of weights that can be used as weights in the weighted sum rule.

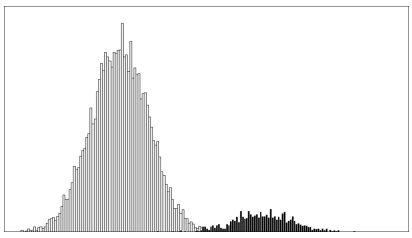

Figure 2. Distribution of positive and negative scores.

C. Validation Accuracy Weighting (VAW)

Another weighting scheme, which we name as validation accuracy, depends on individual recognition rates of each block on validation data. Using training data, a single classifier is trained for each block and each block of a sample in the validation data is classified using the classifier that corresponds to the block of interest. Individual block recognition rates for all samples in the validation data are acquired separately and weights are assigned proportional to the recognition accuracy of each block. If $\text{acc}(k)$ denotes the recognition accuracy for the k^{th} block, weight of the b^{th} block is given as:

$$w_b = \frac{\text{acc}(b)}{\sum_{k=1}^{B} \text{acc}(k)}. \tag{4}$$

IV. Experimental Results and Discussions

In order to evaluate the performance of the proposed weighting schemes, we have conducted several tests on the AR face database [13]. The AR database consists of face images, which are acquired in two different sessions, of 120 subjects. In each session each subject has 13 images (1 neutral, 3 expression change, 3 illumination change, 3 sunglasses and 3 scarf). Therefore, each subject has 26 images. For our tests, we have used the first seven images of the first session for training and the first seven images of the second session for validation. The rest images (12 for each subject) being used for testing, we have conducted decision fusion tests with different weighting schemes. We did not include all the recognition results for brevity, however they can be found in [11]. Apart from the AR database, we have also evaluated our method on the M2VTS database [14]. The recognition results on the M2VTS database followed a similar pattern to the AR database and they can be found in [11].

Table I
Recognition accuracies for EW, FW, SVM-W, VAW weighting schemes on the AR database

	EW	**FW**	**SVM-W**	**VAW**
DCT	75.90%	77.50%	75.62%	75.83%
PCA	78.82%	79.58%	78.82%	79.24%
NNDA	83.75%	84.31%	84.10%	85.69%

Table II
RECOGNITION ACCURACIES FOR FEATURE CONCATENATION

DCT	PCA	NNDA
46.15%	45.71%	48.08%

Table III
ACCURACIES OF CSU FACE IDENTIFICATION EVALUATION SYSTEM

PCA Euclidean	22.15%
PCA Mahalinobis	42.56%
LDA	21.94%
Bayesian ML	23.95%
Bayesian MAP	27.84%

The results presented in Table I shows improvements in the recognition accuracies when a weighting scheme is employed instead of equally weighting the contribution of each block. In all cases, weighting schemes provide slightly higher recognition results.

In addition, following the work of Ekenel and Stiefelhagen [3], we concatenated features extracted from patches and created visual feature vector for face images which are used in recognition (Table II). Applying decision fusion on patch-based face recognition provided higher recognition rates than feature fusion.

We have also compared our recognition accuracy values with the set of algorithms that is provided by the CSU Face Identification Evaluation system [15]. It is a package that contains a standard PCA algorithm, a combination of PCA and LDA algorithms and a Bayesian Intrapersonal/Extrapersonal Image Difference Classifier. The recognition rates of these algorithms are in Table III.

V. CONCLUSION

In this study, we proposed three novel weighting schemes for assigning weights in the weighted sum rule over class-posterior probabilities of patches. With all of these methods, we obtained recognition results slightly higher than using equal weights. Also, combining the outputs of classifiers trained over separate patches is shown to be superior over combining the feature vectors extracted from each patch. Feature fusion and DCT were used for patch-based face recognition in [3]. Decision fusion with equal weights and PCA were proposed in [5]. By using VAW weighting scheme and NNDA method, we obtain the highest recognition accuracy of 85.69% which is significanly higher than those two previous methods.

REFERENCES

[1] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *ACM Comput. Surv.*, vol. 35, no. 4, pp. 399–458, December 2003.

[2] R. Gottumukkal and V. K. Asari, "An improved face recognition technique based on modular PCA approach," *Pattern Recogn. Lett.*, vol. 25, no. 4, pp. 429–436, 2004.

[3] H. Ekenel and R. Stiefelhagen, "Local appearance-based face recognition using discrete cosine transform," in *13th European Signal Processing Conference (EUSIPCO 2005)*, September 2005.

[4] K. Tan and S. Chen, "Adaptively weighted sub-pattern pca for face recognition," *Neurocomputing*, vol. 64, pp. 505 – 511, 2005, trends in Neurocomputing: 12th European Symposium on Artificial Neural Networks 2004. [Online]. Available: http://www.sciencedirect.com/science/article/B6V10-4FB940X-1/2/0d02e8095793af3dac97a219fea09525

[5] Y. Su, S. Shan, X. Chen, and W. Gao, "Patch-based gabor fisher classifier for face recognition," *Pattern Recognition, International Conference on*, vol. 2, pp. 528–531, 2006.

[6] Y. Zhu, J. Liu, and S. Chen, "Semi-random subspace method for face recognition," *Image and Vision Computing*, vol. 27, no. 9, pp. 1358 – 1370, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/B6V09-4VG5HX8-2/2/ffbfbbbfcb86fbfaca1331493ed8afdd

[7] Y. Su, S. Shan, X. Chen, and W. Gao, "Hierarchical ensemble of global and local classifiers for face recognition," *IEEE International Conference on Computer Vision*, pp. 1–8, 2007.

[8] Z. M. Hafed and M. D. Levine, "Face recognition using the discrete cosine transform," *Int. J. Comput. Vision*, vol. 43, no. 3, pp. 167–188, 2001.

[9] M. Kirby and L. Sirovich, "Application of the Karhunen-Loeve procedure for the characterization of human faces," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, no. 1, pp. 103–108, Jan 1990.

[10] X. Qiu and L. Wu, "Stepwise nearest neighbor discriminant analysis," in *Int. Joint Conference on Artificial Intelligence (IJCAI), Edinburgh*, 2005, pp. 829–834.

[11] B. Topcu, "Feature extraction and fusion techniques for patch-based face recognition," Master's thesis, Istanbul, Turkey, 2009.

[12] R. P. W. Duin and D. M. J. Tax, "Classifier conditional posterior probabilities," in *SSPR '98/SPR '98: Proceedings of the Joint IAPR International Workshops on Advances in Pattern Recognition*. London, UK: Springer-Verlag, 1998, pp. 611–619.

[13] A. Martinez and R. Benavente, "The AR face database," CVC, Tech. Rep., 1998.

[14] S. Pigeon and L. Vandendorpe, "The M2VTS Multimodal Face Database (Release 1.00)," in *AVBPA '97: Proceedings of the First International Conference on Audio- and Video-Based Biometric Person Authentication*. London, UK: Springer-Verlag, 1997, pp. 403–409.

[15] D. S. Bolme, J. R. Beveridge, M. Teixeira, and B. A. Draper, "The CSU face identification evaluation system: Its purpose, features, and structure," in *ICVS*, 2003, pp. 304–313.