HYBRID AND PARALLEL FACE CLASSIFIER BASED ON ARTIFICIAL NEURAL NETWORKS AND PRINCIPAL COMPONENT ANALYSIS

Peter Bazanov, Tae-Kyun Kim**, Seok Cheol Kee** and Sang Uk Lee****

* CMC Department, Moscow State University, petr@angel.cs.msu.su
** HCI Lab., Samsung Advance Institute of Technology, {taekyun, sckee}@sait.samsung.co.kr
*** School of Electrical Engineering, Seoul National University, sanguk@sting.snu.ac.kr

ABSTRACT

We present a hybrid and parallel system based on artificial neural networks for a face invariant classifier and general pattern recognition problems. A set of face features is extracted by using the eigenpaxel method, which is based on principal component analysis (PCA) of a group of pixel, that is called a paxel. To classify subjects, multi-layer perceptron neural network (NN)s are trained for each eigenpaxel. These parallel NN kernels provide sage, fast and efficient classification. To combine the results of parallel NNs, a novel judge analyzer is proposed based on bond rating classification and prediction. The proposed judge strategy can detect distinguishable face features even in arguable situations. The proposed method was evaluated on Olivetti and HongIk university (HIU) face databases and it yields that a top recognition rates are 95.5% and 94.11% respectively, which are better results than the previous eigenpaxel and NN approach [1].

1. INTRODUCTION

Our scientific group has pursued researches about hybrid systems based on artificial intelligence for face recognition problems. We have tried to check that the artificial systems like neural networks have a great possibility for successful invariant face recognition. The main issue in our approach is the creation of a hybrid system that combines and emphasizes simple, fast and efficient algorithms for face recognition.

In this paper, we present a face classifier which involves a neural network approach with the feature extraction method of using eigenpaxels, which was originally proposed in [1]. It is because of its simplicity and universality compared to other classification techniques like convolutional networks and neocognitron [2] that this approach is adopted. However, the approach [1], which consists of a NN for all of the eigenpaxel features, has a few great disadvantages in computing efficiency and so on. We improved classification performance by applying the idea that comparable patterns of faces, which are filtered by the same eigenpaxel, are only subjects for classification. That is, one NN for one eigenfilter is constructed to classify patterns which are different in faces but have the same "eigennature". We managed to parallelize their approach and as a result we decreased learning time of NNs and changed representations of pattern database to a distributed mode. Beside the parallel structure of NNs, a powerful judge analyzer is proposed for the combined interpretation of results produced by NNs. The proposed analyzer can detect low-signal features when confidence measures are critical and can make an expert decision even in arguable situations.

The proposed system is modeled by using Stuttgart Neural Network Simulator (SNNS) of a new JavaSNNS 2001 interface [3]. Prototypes of our distributed system have been performed at several MS Window platforms and clusters of 3 Unix servers. Initially our system was oriented to real-time face identification but at this moment our system is more suitable for off-line processing of large image DB, in which a recognition rate is more important than a response time.

2. OVERVIEW

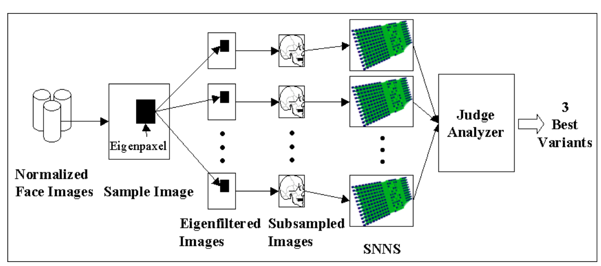

Figure 1. Architecture of the proposed algorithm.

The architecture of the proposed algorithm is illustrated above in Figure 1. All face data are normalized, near-frontal and gray level images. Facial feature vectors are obtained using the eigenpaxel which is based on PCA. The feature extraction method is completely described in [1].

Parallel NN kernels are used for classification networks. The system has one NN for one eigenfilter. A face is recognized by each NN and the results of each NN are combined by using a judge analyzer. The proposed analyzer finally provides 3 best variants. Performance evaluations were done on Olivetti and HIU face DB.

3. FEATURE EXTRACTION

When eigenpaxels are created by PCA of random image blocks, images are filtered by using a subset of eigenpaxels. The largest 4~5 eigenpaxels provided comparable classification results. However, to get more confident results, we utilized the other eigenpaxels. Usually 10~15 eigenpaxels are needed for a sage decision. Figure 2 shows the 15 largest eigenpaxels.

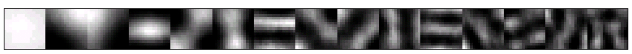

Figure 2. 16x16 Eigenpaxels

At first, face images are convoluted with one eigenpaxel which is shifted as much as predefined percent of overlapping. The all eigenfiltered images are also rectified to the same size and sub-sampled.

In our experiments on two face DB, Olivetti and HIU, we utilized following parameters: 10 eigenpaxels with 16×16 paxel size, 75 % overlapping, 2×2 sub-sampling. Thus, in case of Olivetti DB, original image size is 92×112 pixels and it is reduced into 20×25 pixels after eigenfiltering and 10×12 pixels after sub-sampling. In case of HIU DB, original size is 100×130 pixels and reduced size is 22×29 pixels and 11×14 pixels respectively. Some training examples of Olivetti DB are shown in Figure 3.

Figure 3. Original image, eigenfiltered images with 1st and 10th eigenpaxels.

4. PARALLEL STRUCTURE OF NEURAL NETWORKS

The number of hidden neurons was chosen to provide enough connections for good pattern representation. 60 hidden neurons on Olivetti DB were applied for storing 200 patterns, which are 200 images of 40 subjects of the data set. 100 hidden neurons on HIU DB were chosen for storing 750 patterns. HIU face DB consists of 2500 faces of 250 subjects. These hidden neurons constructed the multi-layer perceptron whose input layer dimension is 10×12 and 11×14 and output layer dimension is 40 and 250 for Olivetti and HIU DB respectively.

The main idea is that one NN is applied for each eigenpaxel. So the same number of NNs as the number of eigenpaxels are parallelized. Advantages are clear :

- Sage classification on the only of comparable patterns: They are different in subjects but have the same nature (Refer to Figure 3).
- Fast teaching of NNs: Each NN learns only a small part of knowledge about faces.
- Novel approaches in creation of a judge analyzer for NNs: We know that the first 4 NNs who correspond to the eigenpaxels with the largest eigenvalues are the most talented for a sage decision making and they should be learned as intensively as possible.
- Parallel structure: It has possibility to simulate algorithms with multi-processor in application domain.
- Small number of training patterns: It has possibility to store and process large DB in a distributed mode.

5. LEARNING BY USING STUTTGART NEURAL NETWORK SIMULATOR

For the creation and learning of the parallel structure of NNs, we have tried 2 interfaces: SNNS with Unix emulation and JAVASNNS. The simulator is very mobile and friendly, so that we could easily perform the algorithm on several computers. One of the advantages of using this simulator is that we can control process during the learning of NNs. By applying a geometric normalization process for input patterns better results were obtained compared to using input images directly. After the normalization and the conversion of eigenfiltered patterns to a SNNS data format, a training procedure is performed.

For the training of NNs we have found 4 major algorithms; *Back propagation, Back propagation with momentum, Quick propagation, Resilient back propagation*. Among them, we got good results on *back propagation with momentum* [5], which is an enhanced version of *Back Propagation* by using a momentum term. The momentum term helps to avoid oscillation problems, which is common with the regular *Back Propagation* when an error surface has a very narrow minimum area. The new weight change, Δw_{ij} is computed by

$$\Delta w_{ij}(t+1) = \eta \delta_j o_i + \alpha \Delta w_{ij}(t)\,, \tag{1}$$

where η is a learning factor, j is an index of current unit, i is an index of predecessor to a current unit j, δ_j is an output error of unit j, o_i is an output of unit i, and α is a constant specifying the influence of the momentum. This adaptation increases learning speed significantly. For other learning algorithms we had some troubles in overall learning time and ease of convergence to local minimum and so on during the learning stage and now it is a subject for further exploration.

Two types of learning for the parallel NN engine are adopted : Brief learning and intensive learning using the following parameters, η=0.1, α=0.05, c=0.05, $d_{\max}$=0.01 for brief and η=0.01, α=0.0005, c=0.01, $d_{\max}$=0.0001 for intensive learning, where η and α are the parameters described in Equation (1) and c describes the amount of flat spot elimination and $d_{\max}$ is a maximum non-propagated error. A learning function is initialized with random weights from [-1,1], and updated with topological order. When an error is approximately less than 0.0001, the learning is stopped. Usually all NNs are trained briefly and the talented NNs (the first 4 NNs) are learned intensively again by continuing to learn the NNs for a long time. Then, results of all trained NNs are combined by using a judge analyzer and an equitable decision is made.

6. NEURAL NETWORK ANALYSER

For the decision of the method how to interpret NN outputs, and how to integrate the results of all NNs, we have explored some analyzers and finally created a judge analyzer which can classify even complex and controversial results. Two types of confidence measure are used for the judge analyzer: a confidential score and a grade sequence. A novel idea for the judge analyzer is proposed. It takes only faces detected by the first four eigenpaxels and makes a decision using the information given by the other eigenpaxels.

The confidential score [2] which compares the maximum neuron output to the second largest output is applied. Before calculating the confidential score, output values are transformed by the softmax algorithm in Equation (2).

$$y_\alpha = \frac{\exp(10w_\alpha)}{\sum_\beta \exp(10w_\beta)}, \tag{2}$$

where w_α is an actual output, and β is the number of all neuron outputs. The confidential score is calculated by

$$conf = y_{\max,1} \cdot (y_{\max,1} - y_{\max,2}), \tag{3}$$

where $y_{\max,1}, y_{\max,2}$ are the largest and second largest transformed outputs. For each NN, the value calculated by Equation (3) is only given to the output which has a maximum value and the rest outputs have zero values. The values are usually summed up through 10 NNs for each output and are used for the proposed judge analyzer. Grade sequence is an integration method of confidences using grades like Figure 4.

Grade sequence : $M = M_1 M_2 ... M_n$

if $conf \geq 0.8$,	then M_i = A,
if $0.6 \leq conf \leq 0.8$,	then M_i = B,
if $0.4 \leq conf \leq 0.6$,	then M_i = C,
if $0.2 \leq conf \leq 0.4$,	then M_i = D,
if $0.1 \leq conf \leq 0.2$,	then M_i = E,
if $0.01 \leq conf \leq 0.1$,	then M_i = F,
if $0.001 \leq conf \leq 0.01$,	then M_i = G,
if $0.0001 \leq conf \leq 0.001$,	then M_i = H,
if $conf \leq 0.0001$,	then M_i = I,

where M_i is a grade corresponding to i th eigenfilter, n is number of eigenpaxels, and *conf* is a confidence for a current neuron output.

Figure 4. Grade sequence for combined analyzer

The proposed grade sequence is novel and it has been inspired from some ideas of bond rating classification and prediction [4]. Representation of results using the grade sequence has less complexity and it is easier to be applied to some expert methods for decision making.

The proposed judge analyzer is implemented by the following procedures:

1) Calculating sum of confidential scores.
2) Grading method.
3) Taking only faces detected by the first 4 NNs.
4) Calculating full length of grade sequence for the detected candidates.
5) Calculating base length (for only 4 NNs.) of grade sequence for the detected candidates.
6) Taking candidates which have maximum sum of confidential scores.

The approach was motivated from the fact that the first eigenfiltered images corresponding to the largest eigenvalue provided the best percent of recognition in all cases and the next 3 filters approximately provided a less but also high percent of recognition. In the step of 3) we managed to detect desired faces with 99% using the first 4 NNs on both Olivetti DB and HIU DB.

7. JUDGE STRATEGY FOR CLASSIFICATION IMPROVEMENT

During classification the best 3 variants are found and are inserted into a group "Best". Initially the faces detected by the first 4 NNs are taken as possible candidates. Sometimes we make a threshold as a grade from “E” to “A” for face authentification problem. For the candidates, two lengths of grade sequences, base and full lengths, are calculated. The length conditions are properly given for base and full length. Candidates are sequentially chosen by the following criterions:

1) If candidates are satisfied in base and full length conditions then they are inserted into a group "Process1".
2) If candidates have the longest full length then they are inserted into a group "Process2".
3) If candidates are satisfied in base length condition then they are inserted into a group "Process3".
4) Then, the selected candidates are sorted by sum of confidential scores in each “Process” group and the ‘A’ grade group and the winners are inserted to the group "Best".

Final candidates are in the group "Best". In our experiments we often had less than 3 variants in the group "Best" and it is because we took only subjects which have a higher confidence than a certain level.

8. EXPERIMENTAL RESULTS

The proposed method was evaluated on Olivetti and HongIk university (HIU) face database. Olivetti database consists of 400 faces of 40 subjects. 200 faces were used for training and the other 200 faces were used for validation. Table 1 shows two results on Olivetti database. Experiment No. 1 shows the result when odd number images are trained and even number images are tested and experiment No. 2 shows the result of the alternative case.

Table 1. Error rates on Olivetti DB.

	Exp. No.1	Exp. No.2
1 Variants	4.5%	4.5%
2 Variants	3.0%	3.0%
3 Variants	2.5%	2.5%

The best fortuitous result which we managed to get is 3.5 %, 1.5 %, 1.5 % error rate for 1, 2, and 3 variants respectively. For briefly learned NNs, we got 6 %, 3.5 %, and 2 % error rate.

HIU face DB consists of 2500 faces of 250 subjects as shown in Figure 5. 750 faces were used for training and 1750 faces were used for validation. Table 2 shows the first brief experiments and the intensive experiments in which central, left and right poses of faces were trained. After the system was updated intensitvely through time, smaller error rates for 1, 2, and 3 variants were obtained.

Figure 5. Left 3 images : training samples, Right 7 images : test samples.

Table 2. Error rates of two experiments on HIU DB

	Brief Exp.	Intensive Exp.
1 Variants	6.06%	5.89%
2 Variants	4.06%	3.89%
3 Variants	3.31%	3.20%

9. CONCLUSION

We have presented a novel algorithm for face recognition problem. Although it has been mentioned about only face recognition here, the proposed algorithm can be applied to the similar pattern recognition problems like face identification and speech recognition. The new hybrid method of artificial neural networks and eigenpaxels was proposed for face invariant classifier. The proposed parallel structure of NNs for each eigenpaxel provided sage, fast and efficient classification. To combine results of multiple NNs, the novel judge analyzer was proposed by taking some ideas from bond rating classification and prediction. The judge strategy could classify faces even in arguable situations.

10. REFERENCES

[1] P. McGuire and G.M.T. D'Eleuterio, “Eigenpaxels and a Neural-Network Approach to Image Classification,” *IEEE Transactions on Neural Networks*, Vol. 12, No. 3, pp. 625~635, May 2001.

[2] S. Lawrence, “Face Recognition: A convolutional neural network approach,” *IEEE Transactions on Neural Networks*, Vol.8, pp. 98~113, No. 1, January 1997.

[3] http://www-ra.informatik.uni-tuebingen.de/SNNS/

[4] H. Okada and M. Narita, “Initializing multilayer neural networks with fuzzy logic,” *International joint conference on Neural Networks*, Vol. 1, pp. 239~244, 1992.

[5] D.J. Swanston, J.M. Bishop, and R.J. Mitchell, "Simple adaptive momentum: new algorithm for training multilayer perceptrons", *Electronics Letters*, pp. 1498~1500, Vol. 30, Issue 18, Sept. 1994.