Hindawi Publishing Corporation
Advances in Artificial Neural Systems
Volume 2011, Article ID 673016, 16 pages
doi:10.1155/2011/673016

Research Article

Applying Artificial Neural Networks for Face Recognition

Thai Hoang Le

Department of Computer Science, Ho Chi Minh University of Science, Ho Chi Minh City 70000, Vietnam

Correspondence should be addressed to Thai Hoang Le, lhthai@fit.hcmus.edu.vn

Received 25 January 2011; Revised 16 June 2011; Accepted 19 July 2011

Academic Editor: Naoyuki Kubota

This paper introduces some novel models for all steps of a face recognition system. In the step of face detection, we propose a hybrid model combining AdaBoost and Artificial Neural Network (ABANN) to solve the process efficiently. In the next step, labeled faces detected by ABANN will be aligned by Active Shape Model and Multi Layer Perceptron. In this alignment step, we propose a new 2D local texture model based on Multi Layer Perceptron. The classifier of the model significantly improves the accuracy and the robustness of local searching on faces with expression variation and ambiguous contours. In the feature extraction step, we describe a methodology for improving the efficiency by the association of two methods: geometric feature based method and Independent Component Analysis method. In the face matching step, we apply a model combining many Neural Networks for matching geometric features of human face. The model links many Neural Networks together, so we call it Multi Artificial Neural Network. MIT + CMU database is used for evaluating our proposed methods for face detection and alignment. Finally, the experimental results of all steps on CallTech database show the feasibility of our proposed model.

1. Introduction

Face recognition is a visual pattern recognition problem. In detail, a face recognition system with the input of an arbitrary image will search in database to output people's identification in the input image. A face recognition system generally consists of four modules as depicted in Figure 1: detection, alignment, feature extraction, and matching, where localization and normalization (face detection and alignment) are processing steps before face recognition (facial feature extraction and matching) is performed [1].

Face detection segments the face areas from the background. In the case of video, the detected faces may need to be tracked using a *face tracking* component. *Face alignment* aims at achieving more accurate localization and at normalizing faces thereby, whereas face detection provides coarse estimates of the location and scale of each detected face. Facial components, such as eyes, nose, and mouth and facial outline, are located; based on the location points, the input face image is normalized with respect to geometrical properties, such as size and pose, using geometrical transforms or morphing. The face is usually further normalized with respect to photometrical properties such illumination and gray scale. After a face is normalized geometrically and photometrically, *feature extraction* is performed to provide effective information that is useful for distinguishing between faces of different persons and stable with respect to the geometrical and photometrical variations. For *face matching*, the extracted feature vector of the input face is matched against those of enrolled faces in the database; it outputs the identity of the face when a match is found with sufficient confidence or indicates an unknown face otherwise.

Artificial neural networks were successfully applied for solving signal processing problems in 20 years [2]. Researchers proposed many different models of artificial neural networks. A challenge is to identify the most appropriate neural network model which can work reliably for solving realistic problem.

This paper provides some basic neural network models and efficiently applies these models in modules of face recognition system. For *face detection module*, a three-layer feedforward artificial neural network with Tanh activation function is proposed that combines AdaBoost to detect human faces so that face detecting rate is rather high. For *face alignment module*, a multilayer perceptron (MLP)

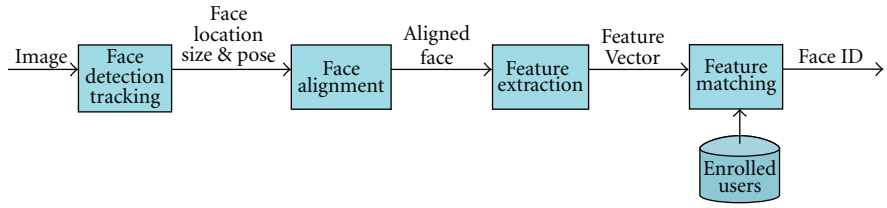

Figure 1: Structure of a face recognition system.

with linear function (three-layer) is proposed, and it creates 2D local texture model for the active shape model (ASM) local searching. For *feature extraction module,* a method for combination of geometric feature-based method and ICA method in facial feature extraction is proposed. For *face matching*, a model which combines many artificial neural networks applied for geometric features classification is proposed. This case study demonstrates how to solve face re cognition in the neural network paradigm. Figure 2 illustrates algorithms for the steps of the face recognition system.

The face detection and alignment steps are conducted on MIT + CMU test set [3] in order to evaluate effectively the performance. Then, the system, which is built from The proposed models, is conducted on CalTech database [4]. Experimental results show that our method performs favorably compared to state-of-the-art methods.

The paper is structured as follows: Section 2 will describe in detail the applying of AdaBoost and artificial neural network for detecting faces. Section 3 will present an ASM method with a novel local texture model, which uses multilayer perceptron (MLP) for ASM local searching. Section 4 will describe a methodology for improving the efficiency of feature extraction stage based on the association of two methods: geometric feature-based method and independent component analysis (ICA) method. Section 5 will present multiartificial neural network (MANN) and MANN application for face matching. The experimental results are presented in Section 6. Conclusions are mentioned in Section 7.

2. AdaBoost and ANN for Face Detection

The face detection processing is the first step of the face recognition system. The step will decide the performance of the system, so it is the most important step of the recognition system. To carry out its efficiently, many researchers have proposed different approaches. In general, there are four groups of face detecting methods [5]: (1) *Knowledge-based methods*; (2) *Invariant feature-based methods*; (3) *Template matching-based methods*; (4) *Machine learning-based methods*.

In this paper, we focus on only machine learning methods because they eliminate subjective thinking factors from human experience. Moreover, they only depend on training data to make final decisions. Thus, if training data is well organized and adequate, then these systems will achieve high performance without human factors.

One of the most popular and efficient learning machine-based approaches for detecting faces is AdaBoost approach [6]. Viola and Jones designed a fast, robust face detection system where AdaBoost learning is used to build nonlinear classifiers. AdaBoost is used to solve the following three fundamental problems: (1) learning effective features from a large feature set; (2) constructing weak classifiers, each of which is based on one of the selected features; (3) boosting the weak classifiers to construct a strong classifier. Viola and Jones make use of several techniques for effective computation of a large number of such features under varying scale and location which is important for real-time performance. Moreover, the cascade of strong classifiers which form cascade tree will make the computation even more efficient. Their system is the first real-time frontal-view face detector. However, their system still has some drawbacks. Since the detection results depend on weak classifiers, the detection results often have many false positives. To decrease the rate of false positives, it is compelled to increase the number of strong classifiers and Haar-like features in cascade tree, but this will cause a significant increase in the performance time, and detection rate can be decreased. Thus, to deal with the issue, we should combine AdaBoost with other machine learning techniques to achieve the same face detecting ratios but with the minimum number of false positives and the running time.

One of the popular methods having the same achievement as well is artificial neural networks (ANNs) [7]. ANN is the term on the method to solve problems by simulating neuron's activities. In detail, ANNs can be most adequately characterized as "computational models" with particular properties such as the ability to adapt or learn, to generalize, or to cluster or organize data, and which operation is based on parallel processing. However, many of the previously mentioned properties can be attributed to non-neural models. A hybrid approach combining AdaBoost and ANN is proposed to detect faces with the purpose of decreasing the performance time but still achieving the desired faces detecting rate.

Our hybrid model is named ABANN. This is the model of combining AB and ANN for detecting faces. In this model, ABs have a role to quickly reject nonface images; then ANNs continue filtering false negative images to achieve better results. The final result is face/nonface.

The selected neural network here is three-layer feed-forward neural network with back propagation algorithm. The number of input neurons T is equivalent to the length of extracted feature vector, and the number of output neurons

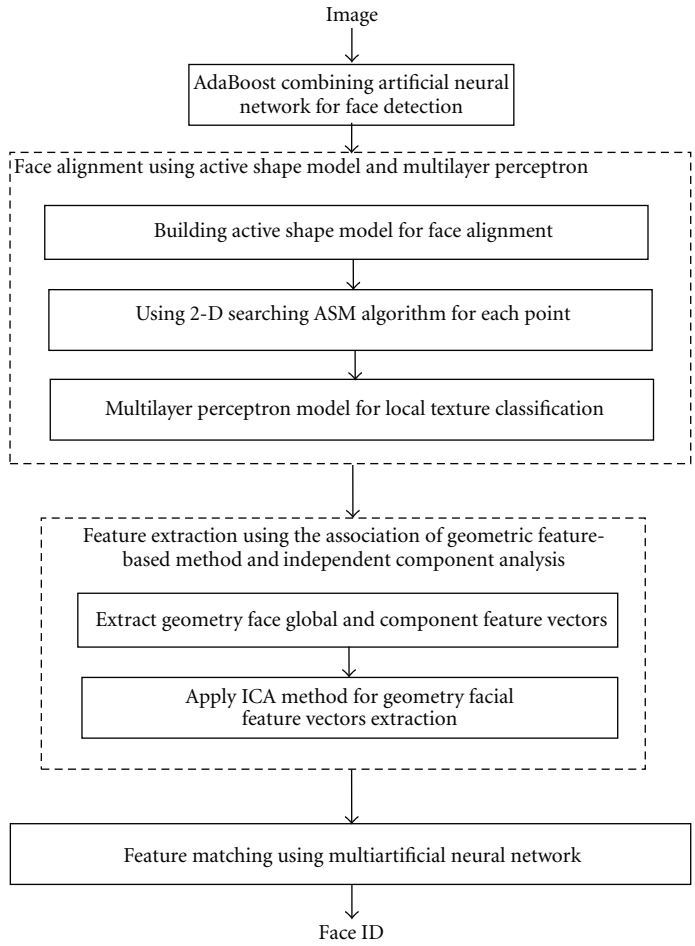

Figure 2: Proposed models for steps of a face recognition system.

is just 1 ($C = 1$), This will return *true* if the image contains a human face and *false* if it does not. The number of hidden neurons H will be selected based on the experiment; it depends on the sample database set of images.

The result image (20×20 pixels) of AB is the input of ANN. The output of the ANN is a real value between -1 (false) and $+1$ (true). The preprocessing and ANN steps are illustrated in Figure 3(b). The original image is decomposed into a pyramid of images as follows: 4 blocks 10×10 pixels, 16 blocks 5×5 pixels, and 5 overlapping blocks 20×6 pixels. Thus, the ANN will have $4 + 16 + 5 = 25$ input nodes. Its goal is to find out important face features: horizontal blocks to find out mouths and eyes, square blocks to find out each of the eyes, noses, and mouths. The system uses one hidden layer with 25 nodes to represent local features that characterize faces well [7]. Its activation function is Tanh function with the learning rate $\varepsilon = 0.3$ [7].

In detail, a model of cascade of classifiers includes many strong classifiers, and ANN is combined with the strong classifiers to be a final strong classifier of the system to achieve better results in Figure 3(a). For example, AB includes 5 strong classifiers, called AB5, which will be combined with ANN, the sixth strong classifier, to be ABANN5.

The image results of the step will be the inputs of the face alignment step. The next section elaborates our proposed method.

3. Local Texture Classifiers Based on Multilayer Perceptron for Face Alignment

The face alignment is one of the important stages of the face recognition. Moreover, face alignment is also used for other face processing applications, such as face modeling and synthesis. Its objective is to localize the feature points on face images such as the contour points of eye, nose, mouth, and face (illustrated in Figure 4).

There have been many face alignment methods. Two popular face alignment methods are active shape model (ASM) and active appearance model (AAM) proposed by Cootes [8]. The two methods use a statistical model to parameterize a face shape with PCA method. However, their feature model and optimization are different. ASM algorithm has a 2-stage loop: in the first stage, given the initial labels, searching for a new position for every label point in its local region which best fits the corresponding local 1D profile texture model; in the second stage, updating the shape parameters which best fit these new label positions.

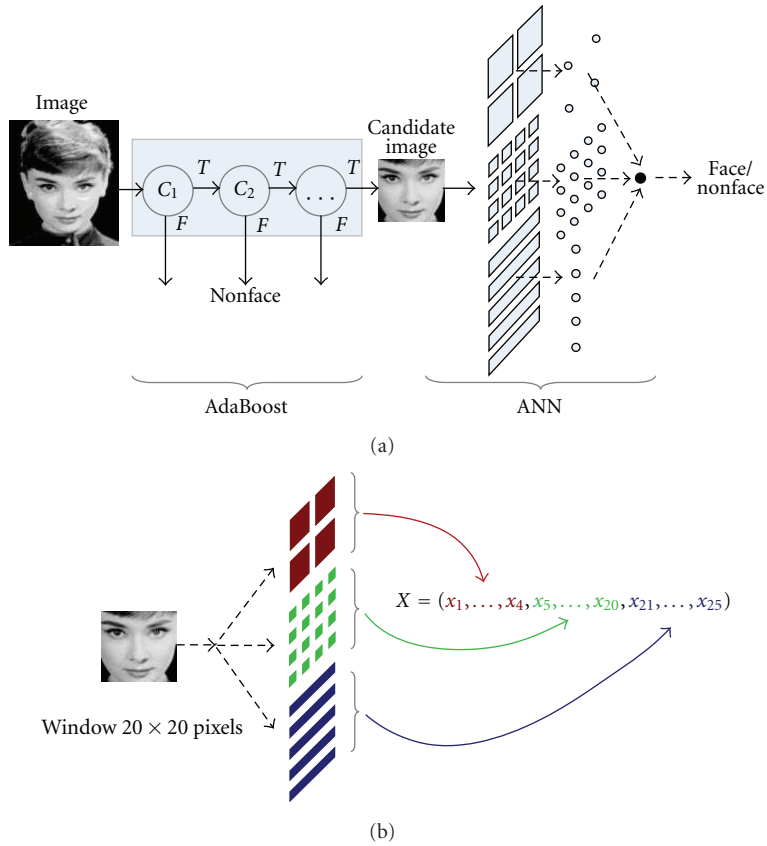

Figure 3: (a) The process of detecting faces of ABANN and (b) input features for neural network.

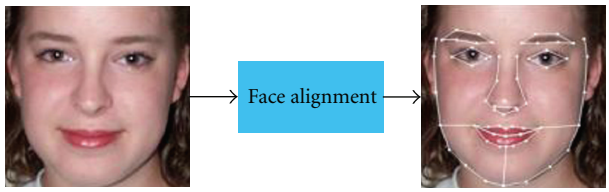

Figure 4: Face alignment.

AAM method uses its global appearance model to directly conduct the optimization of shape parameters. Owing to the different optimization criteria, ASM performs more precisely on shape localization and is quite more robust to illumination and bad initialization. In the paper extent, we develop the classical ASM method to create a new method named MLP-ASM which has achieved better results.

Because ASM only uses a 1D profile texture feature, which is not enough to distinguish feature points from their local regions, the ASM algorithm often fell into local minima problem in the local searching stage. A few representative texture features and pattern recognition methods are proposed to reinforce the ASM local searching, for example, Gabor wavelet [9], Haar wavelet [10], Ranking-Boost [11], and FisherBoost [12]. However, an accurate local texture model to large databases is still unachieved target.

In the next subsection, we present an ASM method with a novel local texture model, which uses multilayer perceptron (MLP) for ASM local searching. MLP is very sufficient for face detecting [13].

3.1. Statistical Shape Models. A face shape can be represented by n points $\{(x_i, y_i)\}$ as a $2n$-element vector, $X = (x_1, y_1, \ldots, x_n, y_n)^T$. Given s training face images, there are s shape vectors $\{X_i\}$. Before we can perform statistical analysis on these vectors, it is important that the shapes represented are in the same coordinate frame. Figure 5 illustrates shape model.

In particular, we seek a parameterized model of the form $X = \text{Model}(b)$ (Figure 6), where b is a vector of parameters of the model. Such a model can be used to generate new vectors, X. If we can model the distribution of parameters, $p_b(b)$, we can limit them so the generated Xs are similar to those in the training set. Similarly, it should be possible to estimate $p_X(X)$ using the model.

To simplify the problem, we first wish to reduce the dimensionality of the data from $2n$ to something more manageable. An effective approach is to apply PCA to the data. The data form a cloud of points in the $2n$-D space. PCA computes the main axes of this cloud, allowing one to approximate any of the original points using a model with fewer than $2n$ parameters. The approach is as follows [1].

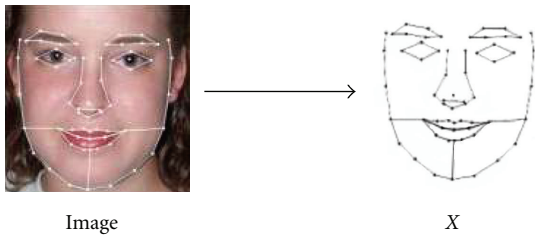

Figure 5: Shape model of an image.

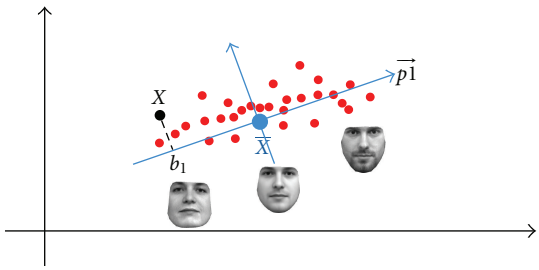

Figure 6: Using PCA to compute statistical shape model.

Step 1. Compute the mean of the data set

$$\overline{X} = \frac{1}{s}\sum_{i=1}^{s} X_i. \tag{1}$$

Step 2. Compute the covariance matrix of the data set

$$S = \frac{1}{s-1}\sum_{i=1}^{s}\left(X_i - \overline{X}\right)\left(X_i - \overline{X}\right)^T. \tag{2}$$

Step 3. Compute the eigenvectors, p_j, and corresponding eigenvalues, λ_j, of the data set S (sorted so $\lambda_j \geq \lambda_{j+1}$).

Step 4. We can approximate X from the training set

$$X \approx \overline{X} + P_s b_s, \tag{3}$$

where $P_s = (p_1 | p_2 | \cdots | p_t)$ (t, the number of modes, can be chosen to explain a given proportion of 98% of the variance in the training data set) and $b_s = (b_1, b_2, \ldots, b_t)$, shape model parameters, given by

$$b_s = P_s^T\left(X - \overline{X}\right), \qquad b_i \in \left\{-3\sqrt{\lambda_i} + 3\sqrt{\lambda_i}\right\}. \tag{4}$$

A real shape $\mathbf{X}$ of images can be generated by applying a suitable transformation $\mathbf{T}$ to the points X:

$$\mathbf{X} = \mathbf{T}\left(\overline{X} + P_s b_s, x_c, y_c, s_x, s_y, \theta\right). \tag{5}$$

This transformation includes a translation (x_c, y_c), a scaling (s_x, s_y), and a rotation (θ).

3.2. ASM Algorithm. Given a rough starting approximation, the parameters of an instance of a model can be modified to better fit the model to a new image. By choosing a set of shape parameters, b_s, for the model, we define the shape of the object in an object-centered coordinate frame. We can create an instance X of the model in the image frame by defining the position (x_c, y_c), orientation θ, and scale (s_x, s_y) parameters. An iterative approach to improve the fit of the instance, $\mathbf{T}(\overline{X} + P_s b_s, x_c, y_c, s_x, s_y, \theta)$ (Figure 7), to an image proceeds as follows.

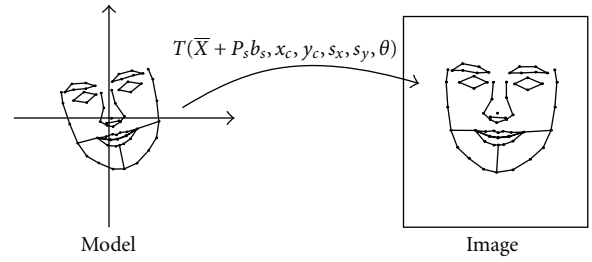

Figure 7: Transformation model into image.

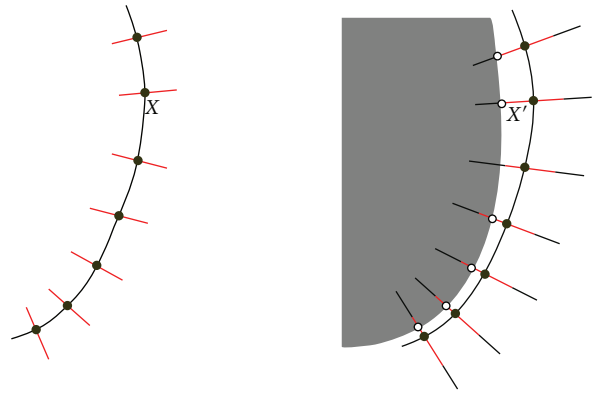

Figure 8: 1D profile texture model.

Step 1. Examine a region of the image around each point of X to find the best nearby match for the points X'. There are some ways to find X'. A popular method, the classical texture model, will be presented in Section 3.3, then our method, the MLP local texture model, will be presented in Section 3.4.

Step 2. Repeat until convergence.

Update the parameters $(b_s, x_c, y_c, s_x, s_y, \theta)$ to best fit to the new found points X' to minimize the sum of square distances between corresponding model and image points:

$$E\left(b_s, x_c, y_c, s_x, s_y, \theta\right) = \left|X' - T\left(\overline{X} + P_s b_s, x_c, y_c, s_x, s_y, \theta\right)\right|^2. \tag{6}$$

Substep 2.1. Fix b_s and find $(x_c, y_c, s_x, s_y, \theta)$ to minimize E.

Substep 2.2. Fix $(x_c, y_c, s_x, s_y, \theta)$ and find b_s to minimize E.

3.3. Classical Local Texture Model. The objective is to search for local match for each point (illustrated in Figure 8). The model is assumed to have The strongest edge, correlation, and statistical model of profile.

Step 1. Computing normal vector at point (x_i, y_i) and calculating tangent vector t,

$$t_x = x_{i+1} - x_{i-1}, \qquad t_y = y_{i+1} - y_{i-1}. \tag{7}$$

Normalize tangent vector t,

$$t_x = \frac{t_x}{|t|}, \qquad t_y = \frac{t_y}{|t|}. \tag{8}$$

Calculate normal vector n,

$$n_x = -t_y, \qquad n_y = t_x. \tag{9}$$

Step 2. Calculate $g(k)$ by sampling along the 1D profile of point (x_i, y_i),

$$G(k) = \text{image}\left[x_i + kn_x, y_i + kn_y\right], \quad k \in [\ldots, -2, -1, 0, 1, 2, \ldots]. \tag{10}$$

To noise images, The average orthogonal to the 1D profile.

$$g(k) = \frac{g_{kl}}{4} + \frac{g_{kc}}{2} + \frac{g_{kr}}{4}. \tag{11}$$

Making the edges of images clear by image derivation, we can select the point at the strongest edge. However, sometimes the true point is not at the strongest edge. We use the local probability model to locate the point. For each point, we estimate the probability density function *(p.d.f)* on the 1D profile from the training data set to search for the correct point. The classical ASM method has some weak points, for example, since PCA did not consider discriminative criterions between positive samples (feature points or true points) and negative samples (nonfeature points, its neighbors), the result of local searching stage often falls into local minima (Figure 9).

To deal with the problem, distinguishing feature points from nonfeature points, which are critical to diminish the effects of local minima problem, we propose the local 2D structure model for each point, which uses MLP trained over a large training set. After training, the model can classify feature points correctly. Multilayer perceptron has been proven to be robust and efficient in face detection [2, 13].

3.4. Multilayer Perceptron Model for Local Texture Classification

3.4.1. Structure of Multilayer Perceptron [2, 13]. A multilayer perceptron (MLP) is a function

$$\hat{y} = \text{MLP}(x, W); \quad x = (x_1, x_2, \ldots, x_n); \quad \hat{y}(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m), \tag{12}$$

W is the set of parameters $\{w_{ij}^L, w_{i0}^L\}$, $\forall i, j, L$.

For each unit i of layer L of the MLP,

Integration:

$$s = \sum_j y_j^{L-1} w_{ij}^L + w_{i0}^L. \tag{13}$$

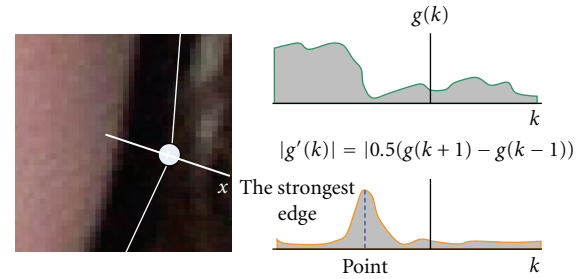

Figure 9: Selecting the feature point at the strongest edge.

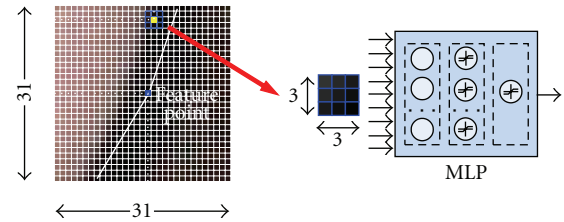

Figure 10: Multilayer perceptron for searching for feature points.

Transfer: $y_j^L = f(s)$, where

$$f(x) = \begin{cases} -1 & x \le -\dfrac{1}{a}, \\ a \cdot x & -\dfrac{1}{a} < x < +\dfrac{1}{a}, \\ 1 & x \ge +\dfrac{1}{a}. \end{cases} \tag{14}$$

On the input layer $(L = 0)$: $y_j^L = x_j$.

On the output layer $(L = \mathbf{L})$: $y_j^L = \hat{y}_j$.

The MLP uses the algorithm of gradient backpropagation for training to update W.

3.4.2. Applying the Multilayer Perceptron for Searching for Feature Points. For each feature point, we define the region of a $[-5,15] \times [-15,15]$ window centered at the feature point. Then, positive samples, feature points, are collected from image points within a $[-1,1] \times [-1,1]$ subwindow at the center, while negative samples, nonfeature points, are sampled randomly out of the sub-window within the region. Then through learning with gradient backpropagation [13], W weights of the MLP are updated, and it outputs a value which is $(+1)$ corresponding to the feature point or (-1) corresponding to the nonfeature point. Figure 10 illustrates MLP for searching for feature points.

The MLP structure for a sub-window has three layers: an input layer, a hidden layer, and an output layer. The input layer has 9 units (input values are the first-order derivation of pixels in the sub-window); the hidden layer has 9 units, and the output layer has one unit (output value $\in \{-1, 1\}$). Such that, the MLP has (9 inputs + 1 bias) $\times$ 9 + 9 + 1 bias = 100 parameters. The MLP uses the transfer function as a linear function with $a = 0.5$ (9) (this is the best fit value

from our experiments over MIT + CMU database [3] and our database).

A local searching procedure will find around the current feature point to be the new feature position, see Algorithm 1.

4. The Association of Geometric Feature-Based Method and Independent Component Analysis in Facial Feature Extraction

One of the most important steps in the face recognition problem is the facial feature extraction. A good feature extraction will increase the performance of face recognition system. Various techniques have been proposed in the literature for this purpose and are mainly classified in four groups. (1) *Geometric feature-based method group*: the features are extracted by using relative positions and sizes of the important components face such as eyes, nose, mouth and other important component of face. The advantage of these methods is the concentration on important components of face such as eyes, nose, and mouth but the disadvantage is not to remain face global structure [14]. (2) *Template-based method Group*: based on a template function and appropriate energy function, this method group will extract the feature of important components of face such as eyes and mouth, or face shape. An image region is the best appropriateness with template (eyes, mouth, etc.) which will minimize the energy [15–17]. Advantages of this group method are using template and determining parameter for important components of face, but disadvantage is not to reflect face global structure. (3) *Color segmentation-based method group*: this group method is based on skin's color to isolate the face [18, 19]. (4) *Appearance-based method group*: The goal of this method group is using linear transformation and statistical methods to find the basic vectors to represent the face. Methods have been proposed in the literature for this aim such as PCA [20] and ICA [21, 22]. In detail, goal of PCA method is to reduce the number of dimensions of feature space, but still to keep principle features to minimize loss of information. PCA method uses second-order statistic (covariance matrix) in the data. However, PCA method has still disadvantages. High-order dependencies still exist in PCA analysis, for example, in tasks as face recognition, much of the important information may be contained in the high-order relationships among the image pixels, not only second order. Therefore, we need to find a method more general than PCA; ICA [23] is a satisfying method. Instead of principle component analysis, ICA uses technique-independent component analysis, an analysis technique that not only uses second-order statistic but also uses high-order statistic (kurtosis). PCA can be derived as a special case of ICA which uses Gaussian source models. In this case, the mixing matrix cannot determine. PCA is not the good method in cases of non-Gaussian source models. In particular, it has been empirically observed that many natural signals, including speech and natural images, are better described as linear combinations of sources with "super-Gaussian" distributions (kurtosis positive). In this case, ICA method is better than PCA method because (1) ICA provides a better probabilistic model of the data. (2) It uniquely identifies the mixing matrix. (3) It finds an unnecessary orthogonal basic which may reconstruct the data better than PCA in the presence of noise such as variations lighting and expressions of face. (4) It is sensitive to high-order statistics in the data, not just to the covariance matrix.

The appearance-based method group has been found the best performer in facial feature extraction problem because it keeps the important information of face image, rejects redundant information, and reflects face global structure.

In many applications of face recognition such as identity authentication for credit card and video surveillance, accuracy of face recognition problem is the best important factor. Therefore, besides principle components, independent components of data and face global structure are kept by PCA and ICA method. Combination of these features with geometric features such as nose, eyes, and mouth in recognition will increase accuracy, confident of face recognition system.

In this section, we present architectures of ICA for face recognition and combining ICA method with geometric feature-based method (GICA) [24], then comparison of GICA method and GPCA method on CalTech database [4]. Since then, practicability of GICA method for face recognition problem is demonstrated.

4.1. Geometric-Face Global Feature Vector. Labeled faces were detected by ABANN20 (Section 2), which will be normalized in a standard size of 30×30 pixels. After standardization, the face images will be represented by vectors $x_{\text{face}} = (x_1, x_2, \ldots, x_{900})^T$.

4.2. Geometric-Face Component Feature Vectors. Section 3 presented a method to align face by MLP-ASM (multilayer perceptron—active shape model). After face alignment, we can detect image regions that contain eyes and mouth, in face images. To define face component feature vectors, we detect two image regions: (a) region 1 contains eyes, which was detected by coordinates of feature points: *left temple* and *right temple* features, *left outer top eye brow* and *left inner top eye brow* features, *right outer top eye brow* and *right inner top eye brow* features, *left eye bottom* and *right eye bottom* features; (b) region 2 contains labeled face without mouth, which was detected by coordinates of feature points: *Left Jaw* and *Right Jaw* features, *Left Outer Top Eye Brow* and *Left Inner Top Eye Brow* features, *Right Outer Top Eye Brow* and *Right Inner Top Eye Brow* features, and *Left Nose Bottom*, *Nose base*, and *Right Nose Bottom*. For instance in Figure 11.

After locating image regions of face image, region 1 and region 2 will be normalized in a standard size of 30×30 pixels. We have vectors that represent eyes image region: $x_{\text{eyes}} = (x_1, x_2, \ldots, x_{900})^T$, and face without mouth image region: $x_{\text{face\_no\_mouth}} = (x_1, x_2, \ldots, x_{900})^T$. These vectors are called "*Geometric-component feature vector*."

4.3. ICA Method for Facial Feature Extraction. Independent component analysis (ICA) minimizes both second-order and higher-order dependencies in the input data and attempts to find the basis along which the data (when projected onto them) are statistically independent. Bartlett et al. [22]

Input shape X $\{(x_i, y_i)\}$
Output new shape X' $\{(x_i', y_i')\}$
  For each point (x_i, y_i) of shape X
    For each sub-window sw' centered at point (x', y')
    of the window centered at the feature point (x_i, y_i).
      (i) Computing MLP (sw', W). If the return value is $(+1)$ then point (x', y') is at the edge.
      (ii) Selecting the nearest point (x', y') to the point
        (x_i, y_i) as the new feature point (x_i', y_i').

Algorithm 1

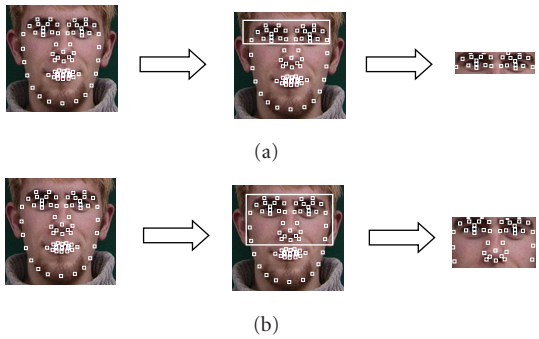

Figure 11: Extraction of face important components: (a) region 1 contains eyes, (b) region 2 contains labeled face without mouth.

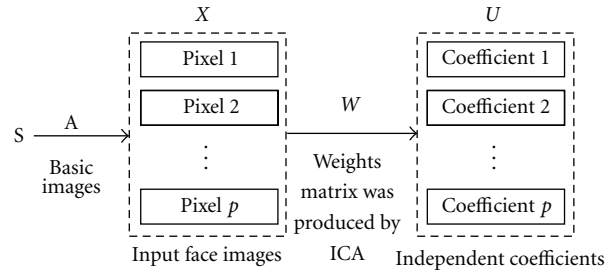

Figure 12: Finding coefficients whose presentation images are independent.

provided two architectures of ICA for face recognition task: architecture I statistically independent basis images, and Architecture II factorial code representation.

In a face recognition problem, our goal is to find coefficients of feature vectors to achieve the most independent in desire. Therefore, in this section, we selected architecture II of ICA method for the face representation. A number of algorithms for performing ICA have been proposed (see [25] for reviews). In this section, we apply FastICA algorithm developed by Hyvärinen and Oja [25] for our experiments.

Architecture II: Statistically Independent Coefficients. The goal in this approach is to find a set of statistically independent coefficients.

We organize the data matrix X so that the images are in columns and the pixels are in rows. Pixels i and j are independent if when moving across the entire set of images, it is not possible to predict the value taken by pixel i based on the corresponding value taken by pixel j on the same image. The goal in architecture I that uses ICA is to find a set of statistically independent basic images. Although basic images found in architecture I are approximately independent, when projecting down statistically independent basic images subspace, feature vectors of each image are not necessarily independent. Architecture II uses ICA to find a representation whose coefficients are used to represent an image in the basic images subspace being statistically independent. Each row of weight matrix W is an image A, an inverse matrix of W, which contains basic images in its columns. Statistically independent coefficients in S will be recovered in columns of U (Figure 12); each column of U contains coefficients for combination of basic images in A to construct images of X.

Architecture II is implemented through the following steps.

Assume that we have n images; each image has p pixels. Therefore, data matrix X has an order of $p \times n$.

(1) Let R be a $p \times m$ matrix containing the first m eigenvectors of a set of n face images in its columns.

(2) Calculating a set of principle components of a set of images in X,

$$C = R^T \times X. \tag{15}$$

(3) The coefficients for linearly combining the basic images in A are determined:

$$U = W \times C. \tag{16}$$

Assume that we have a set of images for testing X_{test}, feature extraction of X_{test} is computed through the following steps: firstly, from X_{test}, we calculate a set of principle components of X_{test} by

$$C_{\text{test}} = R^T \times X_{\text{test}}. \tag{17}$$

Then, a set of feature vectors of X_{test} in the basic images space is calculated by

$$U_{\text{test}} = W \times C_{\text{test}}. \tag{18}$$

Each column of U_{test} is a feature vector corresponding with each image of X_{test}.

Figure 13: First row contains eight eigenfaces corresponding to the highest eight eigenvalues for PCA. Second row first of eight basic images for architecture II ICA.

Firstly, to face representation with ICA method, we apply PCA to project the data into an m-dimensional subspace in order to control the number of independent components made by ICA, and then ICA is applied to the eigenvectors to minimize the statistical dependence of feature vectors in the basic images space. Thus, PCA uncorrelated input data and high-order dependence will remain separated by ICA.

The first row in Figure 13 contains the eight eigenvectors (eight eigenfaces) corresponding to the eight highest eigenvalues of PCA method, and the second row includes the first of eight basic images in architecture II of ICA on CalTech database [4].

4.4. The Association of Geometric Feature-Based Method and ICA in Facial Feature Extraction. After the processed steps of Sections 4.1 and 4.2, we get one vector which represents *geometric-face global feature vector*: $x_{\text{face}} = (x_1, x_2, \ldots, x_{900})^T$ and two vectors which repesent *Geometric-component feature vector*: $x_{\text{eyes}} = (x_1, x_2, \ldots, x_{900})^T$, $x_{\text{face\_no\_mouth}} = (x_1, x_2, \ldots, x_{900})^T$. ICA method was applied to these vectors. Therefore, we get vectors which represent component facial features and global face in ICA subspace.

In detail, vector x_{face} will be mapped into an ICA face feature subspace. Vector representing faces in the face feature subspace is $y_{\text{face}} = (y_1, y_2, \ldots, y_k)^T$, where k is dimensions in the face feature subspace; y_{face} is called "*ICA global feature vector*." Face component feature vectors $x_{\text{eyes}} = (x_1, x_2, \ldots, x_{900})^T$ and $x_{\text{face\_no\_mouth}} = (x_1, x_2, \ldots, x_{900})^T$ will be mapped into ICA component feature subspace; we have $y_{\text{eyes}} = (y_1, y_2, \ldots, y_k)^T$, $y_{\text{face\_no\_mouth}} = (y_1, y_2, \ldots, y_k)^T$. These vectors are called "*ICA component feature vector*."

The combination of these vectors will creates vector which reflects texture and geometric feature of face image. The process of feature extraction in our study was illustrated in Figure 14. y_{comb} is a feature vector of combining the geometric feature-based method and the ICA method. It will be used for face recognition step.

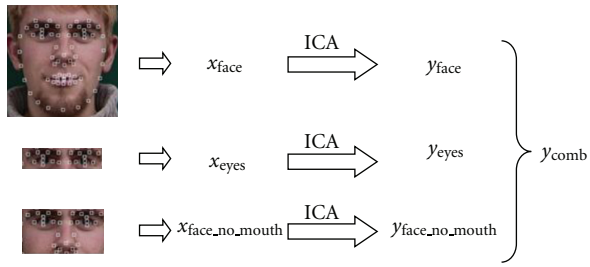

Figure 14: Facial feature extraction by geometric feature-based method combination with ICA method.

5. Multi-artificial Neural Network for Facial Feature Matching

5.1. Multiartificial Neural Network Applys for Pattern Classification. Artificial neural network was successfully applied for face detection and face recognition [26]. Most of the other approaches are to apply ANN for detected face [27, 28]. We proposed the multiartificial neural network (MANN) [29] to apply for pattern and image classification. Firstly, patterns or images are projected to difference spaces. Secondly, in each of these spaces, patterns are classified into responsive class using a neural network called subneural network (SNN) of MANN. Lastly, we use MANN's global frame (GF) consisting some component neural network (CNN) to compose the classified result of all SNN.

5.1.1. The Proposal of MANN Model. Multiartificial neural network (MANN), applying for pattern or image classification with parameters (m, n, L), has m subneural network (SNN) and a global frame (GF) consisting L component neural network (CNN). In particular, m is the number of feature vectors of image, n is the number of feature vector dimensions, and L is the number of classes.

Definition 1. SNN is a 3-layered (input, hidden, and output) neural network. SNN has n (the dimensions of feature vector) input nodes and L (the number classes) output nodes. The number of hidden nodes is experimentally determined. There are m (the number of feature vectors) SNNs in MANN model. The input of the ith SNN, symbol is SNN_i, is the feature vector of image. The output of SNN_i is the classified result based on the ith feature vector of image.

Definition 2. Global frame is the frame consisting L component neural networks which compose the output of SNN(s).

Definition 3. Collective vector kth, symbol R_k $(k = 1 \cdots L)$, is vector joining the output kth of all SNNs. Collective vector

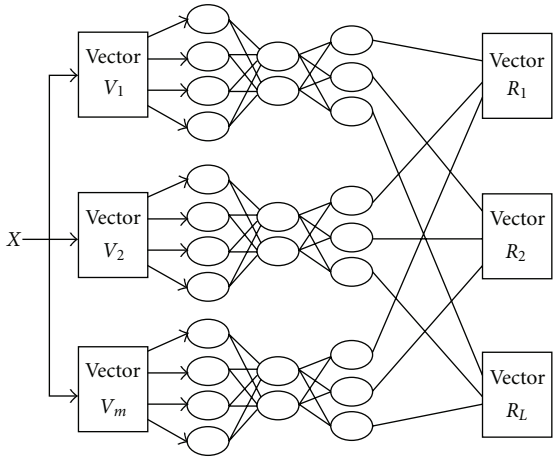

Figure 15: Create collective vector for CNN(s).

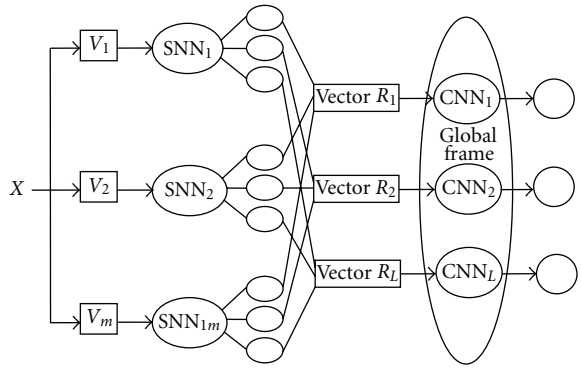

Figure 16: MANN with parameters (m, n, L).

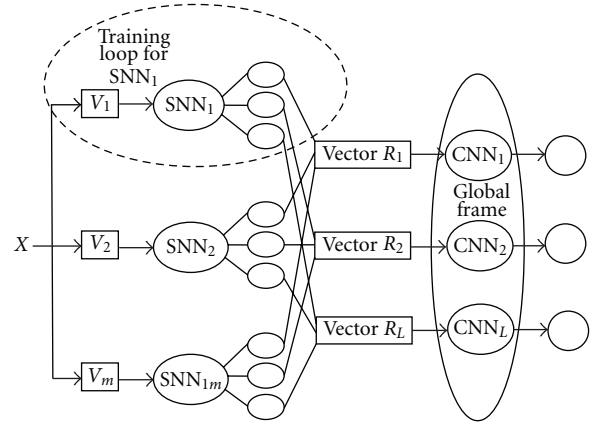

Figure 17: Local training for SNN_1.

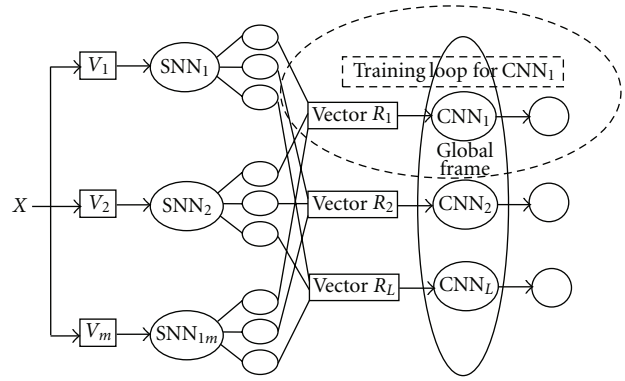

Figure 18: Global training for $CNN_{1.}$.

is m-dimensional vector because there are m SNNs (see Figure 15).

Definition 4. CNN is a 3-layered (input, hidden, and output) neural network. CNN has m (the number of dimensions of collective vector) input nodes and 1 (the number classes) output nodes. The number of hidden nodes is experimentally determined. There are L CNNs. The output of the jth CNN, symbols is CNN_j, gives the probability of X in the jth class (see Figure 16).

5.1.2. The Process of MANN Model. The training process in MANN is separated into two phases. Phase (1) is to train SNN(s) one-by-one called local training. Phase (2) is to train CNN(s) in GF one by one called global training.

In local training phase, we will train the SNN_1 first. After that, we will train $SNN_2, \ldots, SNN_m$ (see Figure 17).

In the global training phase, we will train the CNN_1 first. After that we will train $CNN_2, \ldots, CNN_L$ (see Figure 18).

The classification process of pattern X using MANN is as follows: firstly, pattern X is extracted to m feature vectors. The ith feature vector is the input of SNN_i classifying pattern. All the kth output of all SNNs is joined to create the kth ($k = 1 \cdots L$) collective vector, symbol R_k. R_k is the input of CNN_k. The output of CNN_k is the kth output of MANN. It gives us the probability of X in the kth class. If the kth output is max in all output of MANN and bigger than the threshold, we conclude pattern X in the kth class.

5.2. Face Recognition Using Multiartificial Neural Network

5.2.1. Facial Feature Extraction. After the processed steps of Section 4.4, we get *ICA global feature vector:* $y_{\text{face}} = (y_1, y_2, \ldots, y_k)^T$ and *ICA component feature vector*: $y_{\text{eyes}} = (y_1, y_2, \ldots, y_k)^T$, $y_{\text{face\_no\_mouth}} = (y_1, y_2, \ldots, y_k)^T$, where k is dimensions in the face feature subspace. Figure 19 illustrates the steps of facial feature extraction. Instead of combining these vectors to create y_{comb} (Section 4.4), we used y_{face}, y_{eyes}, and $y_{\text{face\_no\_month}}$ which are inputs of multiartificial neural network (MANN), and outputs of MANN are face identified of candidate face image.

5.2.2. Multiartificial Neural Network for Facial Feature Matching. We have conducted experiments on a CalTech database consisting of 441 labeled faces of 26 people which were detected by ABANN20 (Section 2) and face alignment by MLP_ASM (Section 3). We divide up the database into two parts. Training set contains 120 images and testing set contains 321 images. We repeated our experiments for 10

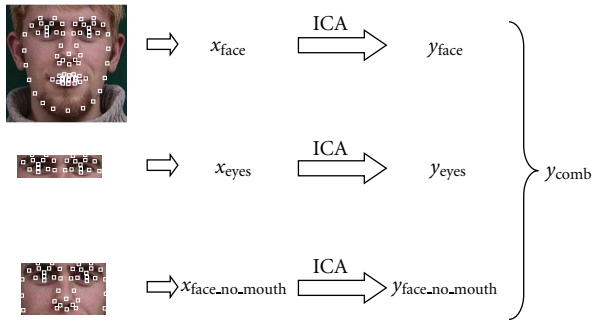

Figure 19: The steps of facial feature extraction.

random divisions of the database, so that every image of the subject can be used for testing. The results were reported on the average performance.

A labeled face is a pattern that is featured by 3 vectors (y_{face}, y_{eyes}, and $y_{face\_no\_mouth}$) which have 100 dimensions ($k = 100$ for ICA) (Section 5.2.1). Face images need to be classified in one of the 26 people (CalTech database). So we apply MANN with parameters ($m = 3$, $n = 100$, $L = 26$) (see Figure 17) for facial feature matching.

Thus, MANN model in this case has three SNNs and one GF consisting of 26 CNNs. The ith ($i = 1 \cdots 3$) feature vector of an image will be processed by SNN_i in order to create the ($L = 26$) dimensional output vector of responsive SNN. To join all the kth ($k = 1 \cdots 26$), The element of these output vectors gets the collective vector R_k. These collective vectors are the input of CNNs. The only one output node of CNN is an output node of MANN.

Our implementation uses backpropagation neural network which has 3 layers with the transfer function that is sigmoid function [2, 30] for SNN and CNN. The number of hidden nodes of SNN_i ($i = 1 \cdots 3$) and CNN_j ($j = 1 \cdots 26$) is experimentally determined from 10 to 100 hidden nodes.

Every SNN_i has $n = 100$ (the dimensions of feature vector) input nodes and $L = 26$ (the number of classes) output nodes. The kth ($k = 1 \cdots 26$) output of the SNN_i is the probability measure which reflects whether the input image is in the kth class.

Every CNN_j has $m = 3$ (the number of feature vectors) input nodes and only one output node. Input of CNN_j is the jth output of all SNNs. It means that CNN_j composes the probability of image in the jth class appraised by all SNNs. Output of CNN_j is the jth output of MANN model. It gives the probability of image in the jth class. It is easy to see that to build MANN model, only neural network technology is used to develop our system.

6. Experimental Results and Discussion

6.1. Face Detection

6.1.1. Database for Experiments. To train the detector, a set of face and nonface training images were used. To create the face training set, we select 11000 face images from 14051 face images of FERET database [31]. With each image which is selected, we cropped the regions which contained face and scaled to a base resolution of 20 by 20 pixels. Some typical face examples are shown in Figure 20.

Table 1: The ANN structure for detecting faces.

Name	Input nodes	Hidden nodes	Output nodes	Learning rate
ANN_FACE	25	25	1	0.3

The nonface subwindows used to train the detector come from 5817 images which were manually inspected and found not to contain any faces. There are about 950 million subwindows within these nonface images. Each classifier in the cascade was trained with the 11000 training faces and 5000 nonface subwindows (also of size 20 by 20 pixels). To train AdaBoost detector, we used open-source Haar training (in OpenCv library) which is created by Lienhart and Maydt [32].

We used Rowley's ANN model [7] for detecting faces presented in Table 1.

Thus, a system is implemented by the three-layer feed-forward ANN with the Tanh activation function (19) and the backpropagation learning algorithm [2]. The system ran on a PC, 2.0 GHz Pentium IV processor, RAM 1 GB. We also used 11000 training faces and 5000 nonface subwindows (also of size 20 by 20 pixels) to train the ANN. It took about 8 hours to train the ANN.

$$\text{Tanh function: } f(x) = \frac{1 - e^{-x}}{1 + e^{-x}}, \quad f(x) \in [-1, 1]. \tag{19}$$

6.1.2. Experimental Results of AdaBoost-Based System. We used the model of cascade of boosted classifiers in which the number of stage classifiers is 20 and 25 stages. We tested the system on the MIT + CMU [3] test set. This database consists of 130 images with 507 labeled faces. Table 2 presents performance of AdaBoost detector.

With a 2.0 GHz Pentium IV processor, RAM 1 GB, the face detector can process a 276-by-343 pixels image in about 0.172 seconds. The scale and step size of slide window are 1.2 and 2, respectively.

6.1.3. Experimental Results of ANN-Based System. We used trained ANN model for detecting faces (Table 1). We also tested the system on the MIT + CMU [3] test set. Table 3 presents the performance of ANN.

6.1.4. Evaluations on AdaBoost and ANN. The experiments prove that AdaBoost and ANN approaches for detecting faces do not achieve good results of performance time and detecting rate yet.

AdaBoost method is one of today's fastest algorithms. However, false face detecting rate is rather high. The cascade of boosted classifier depends on weak classifiers. False images are often false negative. To solve the drawback, there are two solutions. First, we can increase the large number of stage classifiers in order to achieve the desired results. However, increasing the number of both classifiers

Figure 20: Example of face images used for training AdaBoost detector.

Table 2: Performance of detecton on MIT + CMU test set of AdaBoost detector.

Method	Number of stages	Number of Haar-like features used	Face detected	Missed faces (false rejection)	False detections	Detection rates	Average time to process an image (second)
AB20	20	1925	467	40	202	92.11%	0.179
AB25	25	2913	452	55	40	89.15%	0.202

and features too much will decrease the algorithm speed. Second, we can combine AdaBoost with other classification techniques to reject false negative images in order to increase the correctness of the system.

ANN, a strong classification technique, has been used efficiently in the problem of detecting faces. In addition, the performance time is not high. Since then, we suggest a hybrid model of AdaBoost and ANN. On the other hand, we append ANN at the final stage to create a complete hybrid system.

6.1.5. Experimental Results of Hybrid Model of AdaBoost and ANN. AdaBoost and ANN detector are trained The same as in Section 2. These experiments were done on MIT + CMU test set [3]. Table 4 presents The performance of AB ANN detector.

All ABANN20 and ABANN25 get detection rate approximate with AB20 and AB25, respectively, but with less than the number of false detection. ABANN20 gets the detection rate 91.91%; it is approximate with 92.11% of AB20 and higher detection rate of AB25. The number of false positives only is 13. It is very small in comparison with 202 of AB20 and smaller than 40 of AB25. Furthermore, the processing time to process 130 images is 24.576 seconds. It is approximate with the time of AdaBoost detector. In case of ABANN25, it gets a detection rate of 88.76%. This is equal to AB25, but the number of false positives is only 3; this is very small in comparison with 40 of AB25. The processing time of our detector for a 276×343 image is about 0.174 seconds on a P4 1.8 GHz PC.

Figure 21 presents some experimental results on MIT + CMU test set. From the achieved results and theoretical analyses presented in Section 2, we have recognized that the proposed model of associating AdaBoost with ANN is necessary and can be applied in practicality.

We have researched the two popular methods of detecting faces, AdaBoost and ANN, analyzing, and evaluating ones' advantages and disadvantages. From the study, we has recognized that AdaBoost (cascade of boosted AdaBoost) has the fastest performance time; however, the correctness rate is not high (because detection results depend on weak classifiers or Haar-like features); it is proved by the experiments on database CalTech. ANN will reach good verifying results if it has a suitable structure; nevertheless, the detection speed is quite slow due to the complexness of ANN. Hence, in the experiments, we used simple ANN or three-layer feedforward neural network proposed by Rowley. To improve the performance and eliminate its limitations, we have proposed the hybrid model of AdaBoost and ANN (ABANN) for detecting faces. The proposed system has achieved better results of both correctness rate and performance comparing with individual models (AdaBoost or ANN) on database MIT + CMU, and the testing time is insignificant. Since then, we have reached a conclusion that our hybrid model is very efficient and has a practical meaning in the problem of detecting faces.

6.2. Face Alignments. We tested system on the MIT + CMU [3] test set. This database consists of 130 images with 507 labeled faces. We used ABANN20 (Section 2) for face detection and got 466 labeled faces (Table 4). We have conducted experiments on an MIT + CMU database consisting of 450 labeled faces which were chosen from 466 labeled faces that had been detected by ABANN20 (Section 2). They include male and female aging from young to old people, many of which are with exaggerated expressions such as smiles and closed eyes. We randomly chose 300 images for training, and the rest 150 images for testing. The face shape model is made up of 89 feature points that extract with specific groups as follows: face boundary (22 points), right eyebrow (8 points), left eyebrow (8 points), left eye (8 points), right eye (8 points), nose (13 points), and oral (22 points).

For each feature point, an MLP is trained. For comparison, classical ASM was also implemented and trained on the same training set.

6.2.1. Accuracy. The accuracy is measured with point to point error. The feature points were initialized from the face window which was detected by ABANN20 (Section 2).

Table 3: Performance of detecton on MIT + CMU test set of ANN detector.

Method	Face detected	Missed faces (false rejection)	False detections	Detection rates	Average time to process an image (second)
ANN [7]	476	31	17	93.89%	71.56

Table 4: Performance of detection on MIT + CMU test set of ABANN detector.

Name	AdaBoost structure							
	Number of strong classifiers	Number of Haar-like features	ANN structure	Face detected	Missed faces (false rejection)	False detections	Detection rate	Average time to process an image (second)
AB-ANN	20	1925	Rowley's model	466	41	13	91.91%	0.174
AB-ANN	25	2913		450	57	3	88.76%	0.195

Table 5: The average performance time per iteration (a two-stage process).

Algorithm	Classical ASM	MLP ASM
Time per iteration	2 ms	15 ms

Table 6: Percentage accuracy values on the CalTech database (%).

Method	GPCA	GICA
Recognition rate	94.70	96.57

After the alignment procedure, the errors were measured. The average errors of the 89 feature points are compared in Figure 22. The x-axis, which represents the index of feature points, is grouped by organ. It shows that our method outperforms classical ASM; especially, the improvement of our methods is mainly on feature points of mouth and contour.

6.2.2. Efficiency. The average performance time is listed in Table 5. All the tests are carried out on a 2.0 GHz Pentium IV processor, RAM 1 GB. The classical ASM is the fastest since its computation of local texture model is very simple. Our method is a little slower but is still comparable with the classical ASM (Figure 23).

In conclusion, we proposed a robust face alignment algorithm with a local texture model (MLP ASM). Instead of modeling a local feature by 1D profile texture, our classifier is learned from its 2D profile texture patterns against its neighbor ones as a local texture model. The classifier is of a great benefit to the local searching of feature points because of its strong discriminative power. The generality and robustness of the MLP method guarantee the performance. Therefore, compared to existing ones achieving their models in relative small training sets, our method shows potential in practical applications.

6.3. Facial Feature Extraction

6.3.1. Database for Experiments. In this section, we describe our experiments on CalTech face database.

CalTech Database. The database includes Markus Weber's 450 color images at California Institute of Technology. After using AdaBoost + ANN (ABANN20—Section 2) to detect face regions, we get 441 face images of 26 people. 441 labeled faces were aligned by MLP ASM (Section 3). After face alignment, we used 441 of these images to perform GPCA and GICA. In CalTech database, almost all images are frontal faces, and there is not a same illumination. Some images are too dark or too bright. Some images are hidden important components such as eyes. Some images have different face expressions. We divided up database into two parts. Training set contains of 120 images, and testing set contains 321 images. We repeated our experiments for ten random divisions of the database, so that every image of the subject can be used for testing. The results were reported on the average performance. CalTech database is publicly available for research aims at the URL: http://www.vision.caltech.edu/html-files/archive.html. Figure 24 shows sample images from CalTech database.

The recognizer was implemented by the neural network method. Fast artificial neural network is used in our experiment. Fast artificial neural network library (FANN), which is a free open-source neural network library, implements multilayer artificial neural networks in C language and supports for both fully connected and sparsely connected networks. FANN has been used in many studies. FANN implementation includes training step: assume that we have n classes (n different people), training with FANN will create n sets of weights. Each set of weights corresponds to each class (each person).

Testing step: the input is a person's face image (one of the n people mentioned above); this face image was tested with n sets of weights which had been created in the training step. This person belongs to a class which

Figure 21: Output of our face detector on a number of test images from the MIT + CMU test set.

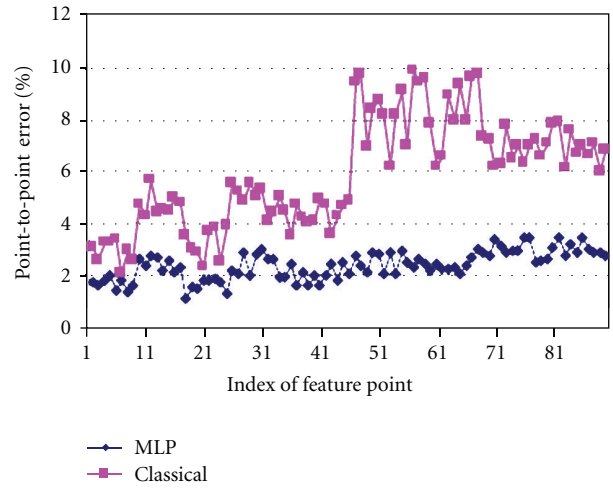

Figure 22: Comparison of classical ASM and MLP ASM.

Table 7: Percentage accuracy values on the CalTech database (%).

Method	SNN1 (y_{face})	SNN2 (y_{eyes})	SNN3 ($y_{face\_no\_mouth}$)	GICA	Average	MANN
Precision	93.25	91.11	94.75	96.57	96.84	98.91

corresponding to the set of weights makes the biggest output. FANN is publicly available for research aims at the URL: http://leenissen.dk/fann/.

6.3.2. Results on the CalTech Database. Table 6 reports the results on CalTech database of two different algorithms applied to the face recognition. The recognition rate for GICA method shows a better recognition rate.

The average number of principal components for PCA representation of *global feature vector* and *component feature vector* was 100 ($k = 100$). The average number of independent components for ICA representation of *global feature vector* and *component feature vector* was 100 ($k = 100$).

The cumulative match score versus rank curve is used to show the performance of each method (Figure 25). Here, cumulative match score and rank are the percentage accuracy that can be achieved by considering the first k biggest output (corresponds with k classes in database) of FANN. The rank is a reliability measure and is very important for video-surveillance applications in uncontrolled environments. Even in this case, the GICA method gives a sharp improvement of the performance in comparison with GPCA method.

A technique for automatic facial feature extraction based on the geometric features of human face and ICA method is presented. With applying neural network method in the recognition step, this paper makes comparisons and evaluations about GPCA and GICA methods on CalTech database (containing 450 images). Through experiment, we notice that GICA method will have good results in cases of face input with the suitable brightness and position mentioned in Section 6.3.1. In future, we continue experiment with other face database so that it demonstrates The practicability of GICA method in face recognition system.

6.4. Recognition Results of System. We evaluated our proposed system with an MANN model with parameters ($m = 3$, $n = 100$, $L = 26$) by all steps in Section 5.2. We compare our propose MANN model with selected method (choose only one subneural network result), average combination method, and GICA (Section 4). The average experimental classified result uses SNN, average combination method, and MANN in the same image set in Table 7.

In this research, the classification result in the CalTech database shows that the proposed model MANN improves the classification result versus the selection and average combination method and GICA (y_{comb} in Section 4).

7. Conclusions

This paper presented novel models for all steps of the recognition of human faces in 2-dimentional digital images. For *face detection module*, a model to combine three-layer

Figure 23: Instances from experimental results.

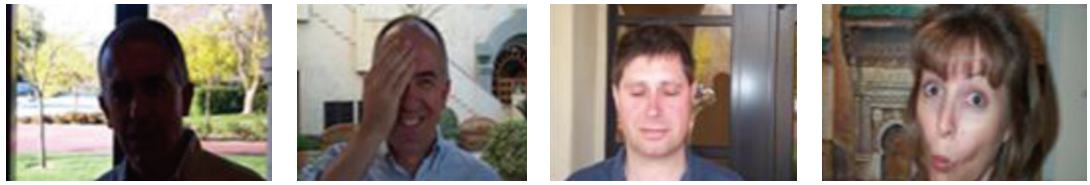

Figure 24: Samples of face images from CalTech database.

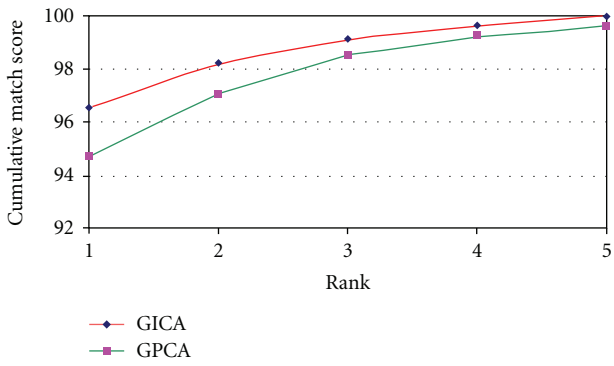

Figure 25: Rank curves on the CalTech database. Reported results show that the GICA method produces a more reliable system.

feedforward artificial neural network and AdaBoost was presented for detecting human faces. The experiments were done on a difficult face detection database which has been widely studied (MIT + CMU database). The results show that ABANN not only gets approximate detection rate and processing time AdaBoost detector but also minimizes false detections. ABANN had solved the drawbacks of AdaBoost and ANN detector. For *face alignment module*, an MLP-ASM model was presented; multilayer perceptron (MLP) was used as a 2D local texture model for the active shape model (ASM) local searching. Comparison of classical ASM and MLP ASM was done on MIT + CMU database which shows the feasibility of MLP ASM model. For *feature extraction module*, a method (GICA) for combination of geometric feature based method and ICA method in facial feature extraction was presented. GICA was comparable with the GPCA on the same database (CalTech database) which indicates the usefulness of GICA. For *face matching*, a model, which combines many artificial neural networks for pattern recognition (multiartificial neural network (MANN)) [29], was applied for ICA-geometric features classification. Comparison with some of the existing traditional techniques in the face recognition rate on the same database (CalTech database) shows the feasibility of MANN model.

References

[1] S. Z. Li and A. K. Jain, *Handbook of Face Recognition*, Springer, New York, NY, USA, 2004.

[2] C. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, NY, USA, 2006.

[3] CBCL Database #1, Center for Biological and Computational Learning at MIT and MIT, http://cbcl.mit.edu/software-datasets/FaceData2.html.

[4] Markus Weber, Frontal Face Database, California Institute of Technology, 1999, http://www.vision.caltech.edu/html-files/archive.html/.

[5] M. H. Yang, D. J. Kriegman, and N. Ahuja, "Detecting faces in images: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34–58, 2002.

[6] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 511–518, December 2001.

[7] H. A. Rowley, *Neural Network Based Face Detection*, Neural network Based Face Detection, School of Computer Science, Computer Science Department, Carnegie Mellon University , Pittsburgh, Pa, USA, 1999.

[8] T. F. Cootes, "Statistical models of appearance for computer vision," http://www.isbe.man.ac.uk/~bim/refs.html/.

[9] F. Jiao, S. Li, H.-Y. Shum, and D. Schuurmans, "Face alignment using statistical models and wavelet features," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2003.

[10] F. Zuo and P. H. N. D. With, "Fast facial feature extraction using a deformable shape model with haar-wavelet based local texture attributes," in *Proceedings of the International Conference on Image Processing (ICIP '04)*, pp. 1425–1428, October 2004.

[11] S. Yan, M. Li, H. Zhang, and Q. Cheng, "Ranking prior likelihood distributions for Bayesian shape localization framework," in *Proceedings of the 8th IEEE International Conference on Computer Vision*, pp. 51–58, October 2003.

[12] J. Tu, Z. Zhang, Z. Zeng, and T. Huang, "Face localization via hierarchical CONDENSATION with Fisher Boosting feature selection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '04)*, pp. I719–I724, July 2004.

[13] S. Marcel, "Artificial neural network for pattern recognition: application to face detection and recognition," 2004, http://www.idiap.ch/~marcel/.

[14] T. Kawaguchi, D. Hidaka, and M. Rizon, "Detection of eyes from human faces by Hough transform and separability filter," in *Proceedings of the International Conference on Image Processing*, vol. 1, no. 2000, pp. 49–52, Vancouver, Canada, 2000.

[15] A. L. Yuille, D. S. Cohen, and P. W. Hallinan, "Feature extraction from faces using deformable template. Computer vision and pattern recognition," in *Proceedings of the IEEE Computer Society Conference*, pp. 104–109, San Diego, CA , USA, June 1989.

[16] L. Zhang, "Estimation of the mouth features using deformable templates," in *Proceedings of the International Conference on Image Processing*, vol. 3, pp. 328–331, Santa Barbara, CA , USA, October 1997.

[17] P. Kuo and J. Hannah, "An improved eye feature extraction algorithm based on deformable templates," in *Proceedings of the IEEE International Conference on Image Processing (ICIP '05)*, pp. 1206–1209, September 2005.

[18] S. L. Phung, A. Bouzerdoum, and D. Chai, "Skin segmentation using color and edge information. Signal processing and its applications," in *Proceedings of the 7th International Symposium*, vol. 1, pp. 525–528, July 2003.

[19] T. Sawangsri, V. Patanavijit, and S. Jitapunkul, "Segmentation using novel skin-color map and morphological technique," in *Proceedings of the World Academy of Science, Engineering and Technology*, vol. 2, January 2005.

[20] M. Turk and A. Pentland, "Face recognition using eigenfaces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–591, 1991.

[21] B. A. Draper, K. Baek, M. S. Bartlett, and J. R. Beveridge, "Recognizing faces with PCA and ICA," *Computer Vision and Image Understanding*, vol. 91, no. 1-2, pp. 115–137, 2003.

[22] M. S. Bartlett, J. R. Movellan, and T. J. Sejnowski, "Face recognition by independent component analysis," *IEEE Transactions on Neural Networks*, vol. 13, no. 6, pp. 1450–1464, 2002.

[23] P. Comon, "Independent component analysis—a new concept?" *Signal Processing*, vol. 36, no. 3, pp. 287–314, 1994.

[24] T. T. Do and T. H. Le, "Facial feature extraction using geometric feature and independent component analysis," in *Proceedings of the Pacific Rim Knowedge Acquisition Workshop (PKAW '08)*, Hanoi, Vietnam, December 2008, (Revised Selected Papers) in Knowledge Acquisition: Approaches, Algorithms and Applications, Lecture Notes in Artificial Intelligence, Springer, Berlin, Germany, pp.231–241, 2009.

[25] A. Hyvärinen and E. Oja, "Independent component analysis: algorithms and applications," *Neural Networks*, vol. 13, no. 4-5, pp. 411–430, 2000.

[26] O. A. Uwechue and A. S. Pandya, *Human Face Recognition using Third-Order Synthetic Neural Networks*, The Springer International Series in Engineering and Computer Science, Springer, 1st edition, 1997.

[27] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 23–38, 1998.

[28] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: a convolutional neural network approach," *IEEE Transactions on Neural Networks, Special Issue on Neural Networks and Pattern Recognition*, vol. 8, no. 1, pp. 98–113, 1997.

[29] T. H. Le, N. T. D. Nguyen, and H. S. Tran, "Landscape image of regional tourism classification using neural network," in *Proceedings of the 3rd International Conference on Communications and Electronics (ICCE '10)*, Nha Trang, Vietnam, August 2010.

[30] L. H. Thai, *Building, development and application, some combination model of neural network (NN), fuzzy logic(FL) and genetics algorithm (GA)*, Ph.D. thesis, Natural Science University, HCM City, Vietnam, 2004.

[31] P. J. Phillips, H. Moon, P. J. Rauss, and S. A. Rizvi, "The FERET evaluation methodology for face-recognition algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1090–1104, 2000.

[32] R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," in *Proceedings of the International Conference on Image Processing*, September 2002.