11

Applications of Artificial Neural Networks to Facial Image Processing

Thai Hoang Le
Department of Computer Science, HCMC University of Science, HCM City Vietnam

1. Introduction

During the past 20 years, artificial neural networks was successfully applied for solving signal processing problems. 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 chapter provides some basic neural network model and efficiently applying these models in facial image processing problem. In detail, three techniques : a hybrid model of combining AdaBoost and Artificial Neural Network (AANN) to detect human faces, a local texture model based on Multi Layer Perceptron (MLP) for face alignment and a model which combines many Neural Networks applied for facial expression classification are present. This case study demonstrates how to solve face recognition in the neural network paradigm. Each of these techniques is introducted as follows:

Technique 1 - an approach to combine adaBoost and artificial neural network for detecting human faces: The human face image recognition is one of the prominent problems at present. Recognizing human faces correctly will aid some fields such as national defense and person verification. One of the most vital processing of recognizing face images is to detect human faces in the images. Some approaches have been used to detect human faces. However, they still have some limitations. In the research, some popular methods, AdaBoost, Artificial Neural Network (ANN) were considered for detecting human faces. Then, a hybrid model of combining AdaBoost and Artificial Neural Network was applied to solve the process efficiently. The system which was build from the hybrid model has been conducted on database CalTech. The recognition correctness is more than 96%. It shows the feasibility of the proposed model.

Technique 2 - local texture classifiers based on multi layer perceptron for face alignment: Local texture models for face alignment have been proposed by many different authors. One of popular models is Principle Component Analysis (PCA) local texture model in Active Shape Model (ASM). The method uses local 1-D profile texture model to search for a new position for every label point. However, it is not sufficient to distinguish feature points from their neighbours; i.e., the ASM algorithm often faces local minima problem. In the research, a new local texture model based on Multi Layer Perceptron (MLP) was proposed. The model is trained from large databases. The classifier of the model significantly improves accuracy and robustness of local searching on faces with expression variation and ambiguous contours. Achieved experimental results on CalTech database show its practicality.

Technique 3 - facial expression classification based on multi artificial neural network: In recent years, image classification and facial expression classification have received much attention. Many approaches are suggested to solve these problems with aiming to increase efficient classification. One of famous suggestions is described as first step, project the pattern or image to different spaces; second step, in each of these spaces, patterns are classified into responsive class and the last step, combine the above classified results into the final result. The advantages of this approach are to reflect fulfill and multiform of image classified. Based on these advantages, classification system improves its precision. In the research, a model which combines many Neural Networks was developed and applied for the last step. This model evaluates the reliability of each space and gives the final classification conclusion. Our model links many Neural Networks together, so we call it Multi Artificial Neural Network (MANN). The proposal model was applied for 6 basic facial expressions on JAFFE database consisting 213 images posed by 10 Japanese female models.

2. An approach to combine adaBoost and Artificial Neural Network for detecting human faces (Thai et al., 2008)

Face recognition is the problem to search human faces in large image database. 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. The face recognition includes the processing in figure 1.

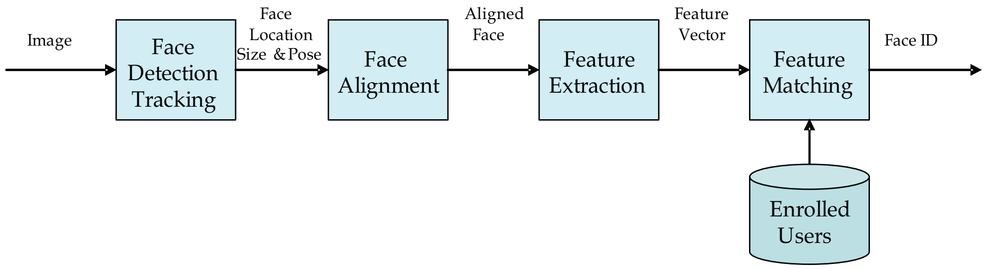


Fig. 1. Structure of a face recognition system

Thus, the face detection processing is the first step of the face recognition system. The step will decide the efficiency of the system, so it is the most important step of the recognition system. Hence, the study only focuses on this step. To carry out it efficiently, many researchers have proposed different approaches. In general, according to Yang et al., 2002, there are four groups of face detecting methods.

Knowledge based methods: these methods are based on sets of rules which have been built from experts on standard face structures. These rules are based on relationships between face features. The methods are mostly used to locate positions of faces. Typical researchers are Kanade et al. , 1973, G.Yang et al. , 1994, and Kotropoulos et al., 1997.

Invariant feature based methods: these methods focus on finding invariant features which always exist in every condition: changes in facial appearance, lighting and expression. Then these features are used only to locate positions of faces. Works which belong in these approaches are K.C.Yow et al., 1997, T.K.Leung et al., 1995.

Template matching based methods: In the approaches, to describe faces or individual face features, face templates would be stored. Detecting faces is based on the correlation between

input images and the stored templates. These methods are used both to locate and detect faces. Some typical researchers are Craw et al., 1992 and A.Lanitis et al., 1995.
Machine learning based methods: in contrast to template matching based methods, models of the methods will learn from training sets of image. After that these models will be used to detect faces. These approaches are used only to detect faces. There are some machine learning models based on these methods such as Eigenface (M.Turk et al., 1991), Probability Distribution Based Model (K.Sung et al., 1998), Artificial Neural Network (H.Rowley, 1998), SVM (E.Osuna et al., 1997), Bayes Classification (H. Schneiderman et al., 1998), Hidden Markov Model (A.Rajagopalan et al., 2005), Reinforcement Learning Model: AdaBoost (Viola et al., 2001); FloatBoost (Stan Z.Li et al., 2004).
In the study, machine learning methods is only focused 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 enough, then these systems will achieve high performance without human factors.
A method of detecting face is to classify the pattern in the sub window as either face or nonface. The classifier is trained by the training set which include face images and nonface images taken under different conditions or extracted in the process of running the program. Face images for training are a part of faces, including left and right eyes, noses and mouths in Figure 2a; nonface images for training do not contain any part of faces in Figure 2b. Training 20x20 images are used for training classifiers. The trained classifiers are able to classify a part of image as face or nonface. In detail, a subwindow 20x20 will be slid on a full image (resized to 120x90). The subwindow is verified by the classifiers to contain a face or not. If the region contains a face, the program will locate it in Figure 3.

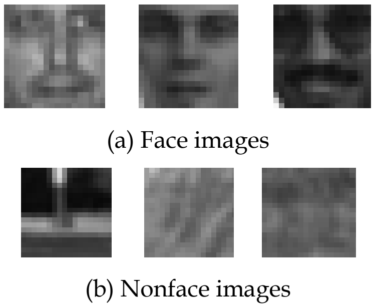


Fig. 2. Images for training classifiers

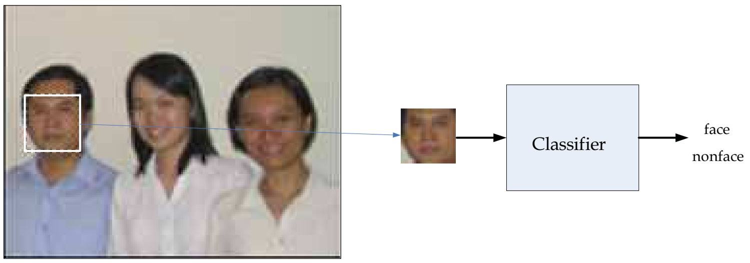


Fig. 3. Classifier 's process for detecting face

Building the classifier is quite feasible because pixels on face images have high correlation to totally describe face structures while ones on nonface images have not the characteristic.
One of the most popular and efficient learning machine based approaches for detecting faces is AdaBoost approach (P. Viola et al., 2001). Viola et al. 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; and (3) boosting the weak classifiers to construct a strong classifier. Weak classifiers are based on Haar-like features which will be presented in Section 2.1. Viola et al. make use of several techniques for effective computation of a large number of such features under varying scale and location which is important for realtime performance. Moreover, the simple-to-complex cascade of classifiers makes the computation even more efficient, which follows the principles of pattern rejection and coarse-to-fine search. Their system is the first realtime frontal-view face detector, and it runs at about 14 frames per second on a 320×240 image (M. H. Yang et al., 2002). However, to achieve high ratios of detecting faces, we must increase the number of classifiers and Haar-like features. It will cause a significant increase in the performance time. Thus to deal with the issue, we should combine AdaBoost with other machine learning techniques to still achieve both the same face detecting ratios and the minimum performance time. One of the popular methods having the same achievement as well is Artificial Neural Networks (H. A. Rowley et al., 1999).
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 previous mentioned properties can be attributed to non-neural models.
In the study, a hybrid approach combining AdaBoost and ANN was suggested to detect faces with the purpose of decreasing the performance time but still achieving the desired faces detecting rate. Section 2.1 is structured as follows. Subsection A will describe in detail on applying AdaBoost and Artificial Neural Network for detecting faces. Subsection B will present the combination model of two previous methods to efficiently solve the face detecting problem and describe experimental result of the proposed system (a software built from the model) conducted on standard databases. Subsection C will give our own evaluations and discussions on the proposed model.

2.1 AdaBoost and ANN model for solving the problem

A. AdaBoost

1. Overview of AdaBoost

AdaBoost is a Boosting algorithm. It originates the PAC learning. It is proved that a combination of weak classifiers will construct a strong classifier. AdaBoost is very efficient because it combines simple statistical learners while reducing significantly not only the training error but also the more vague generalization error.

2. Applying AdaBoost for detecting faces

In detecting faces, AdaBoost based approaches has two main steps. In the first step, strong classifiers will be constructed from weak classifiers. Since then in the second step, the strong classifiers will be combined sequentially to create a cascade of boosted classifier.

a. Building strong classifier

Input: Training data D = n Where $x_k = \left(x_k^1, x_{k,}^2, \ldots, x_k^m\right) \in X$ is a feature vector, $y_k \in Y = \{-1, +1\}$ is a corresponding label (+1 corresponding to face, -1 corresponding to nonface). T weak classifiers $h_j: X \rightarrow \{-1, +1\}$.
We use AdaBoost algorithm to build a strong classifier
Output: The final strong classifier is

$$H(x) = \text{sign}\left(\sum_{j=1}^{j=T} \alpha_j h_j(x)\right) \tag{1}$$

Weak classifiers $h_j(x)$ are simple Haar-like features and a large set of very simple "weak" classifiers that use a single feature to classify the image region as face or nonface.
Each feature is described by the template (shape of the feature), its coordinate relative to the search window origin and the size (scale factor) of the feature. Viola et al. proposed 4 basic templates of scalar features for face detection. Stewart Taylor used 8 different templates and Rainer Lienhart extended to a set of 14 templates (S. Z. Li et al., 2005). Each feature consists of two or three connected "black" and "white" rectangles, either up-right. The Haar-like feature's value is computed as a weighted sum of two components: The pixel sum over the black rectangle and the sum over the whole feature area. The weights of these two components are of opposite signs and for normalization; their absolute values are inversely proportional to the areas. In real classifiers, hundreds of features are used, so direct computation of pixel sums over multiple small rectangles would make the detection very slow. However, Viola et al. suggested a clever method to compute the sums very fast. First, an integral image, Summed Area Table (SAT), is computed over the whole image I, where the pixel sum over a rectangle $r = \{(x,y), x_0 \leq x < x_0+w, y_0 \leq y < y_0+h\}$ can then be computed using SAT by using just the corners of the rectangle regardless of size RecSum(r) = SAT(x_0+w, y_0+h) − SAT(x_0+w, y_0) − SAT(x_0, y_0+h) + SAT(x_0, y_0). This is for up-right rectangles. For rotated rectangles, a separate "rotated" SAT must be used. The computed feature value $x_j = w_{j0}\text{RecSum}(r_{j0}) + w_{j1}\text{RecSum}(r_{j1})$ is then used as input to a very simple decision tree classifier that usually has just two nodes which are computed in Eq. (2) or three nodes calculated in Eq. (2)

$$h_j = \begin{cases} +1 & x_j \leq \theta_j \\ -1 & x_j > \theta_j \end{cases} \tag{2}$$

$$h_j = \begin{cases} +1 & \theta_{j0} \leq x_j < \theta_{j1} \\ -1 & xj < \theta_{j0} \\ -1 & xj \geq \theta_{j1} \end{cases} \tag{3}$$

Where the response +1 means the face, and −1 means the non-face. Every such classifier, called a weak classifier, is not able to detect a face; rather, it reacts to some simple features in the image that may relate to the face.
However, to achieve good results, an AdaBoost based system need a huge number of features. For example, for a subwindow of size 20x20, there can be tens of thousands of such features for varying shapes, sizes and locations. Since then, this significantly decreases the performance speed of the face detecting system. Moreover, final classifier correctness

depends on the correctness of weak classifiers (Haar-like features). Thus the performance effectiveness is not high.

b. Building cascade of boosted classifier

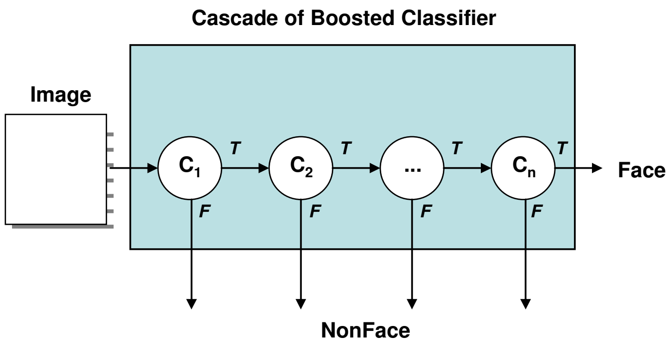


Fig. 4. Structure of cascade of boosted classifier

Most good classifiers need much time to have classification results because they must consider the great number of features of patterns. The cascade structure of strong or boosted classifiers has been suggested in order to reduce performance time, false alarm rates for the classifier (figure 4). The cascade tree has some stages; each stage is a stage classifier. During the detection stage, each pattern is analyzed sequentially by each of the stage classifier they may reject it or let it go through. During the training stage, each stage classifier is trained by false negative patterns of the previous stage. It means that it will learn difficult background patterns. Thus the combination of classifiers in cascade will decrease false alarm rate. With this structure, the classifier can easily recognize background patterns and reject them with first stages. Hence it solves two problems which are complexity and performance time. In summary, the cascade structure has partly improved the performance time, but the detection rate still depends on weak classifiers.

B. ANN

1. Feedforward ANN

There are many prototypes of ANNs. However, this section only concerns feedforward ANN. Feedforward ANN is considered as a mapping between the sets of input and output values. It plays a role of a function f that maps the input set I into the output set O, i.e.: f : I $\rightarrow$ O or y = f(x), where y $\in$ O , and x $\in$ I.

Each output neuron is real value $\in$ [-1, 1] or [0, 1] depend on the transfer function of ANN. A transfer function which can be linear or nonlinear is used to transform information for each neuron of ANN. Some popular transfer functions are Tanh, Sigmoid and Linear.

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

Sigmoid function: $$\mathrm{f}(\mathrm{x})=\frac{1}{1+\mathrm{e}^{-\mathrm{x}}}, \mathrm{f}(\mathrm{x})\in[0,1]$$

Linear function: $$f(x)=\begin{cases}1 & 1/a < x\\ \alpha x & -1/a \le x \le 1/a,\ f(x)\in[-1,1]\\ -1 & x < -1/a\end{cases}$$

The decision object will belong to the class which has maximum neuron. This network is trained by a back propagation algorithm (C. Bishop et al., 2006).

2. Applying the network to verify human faces

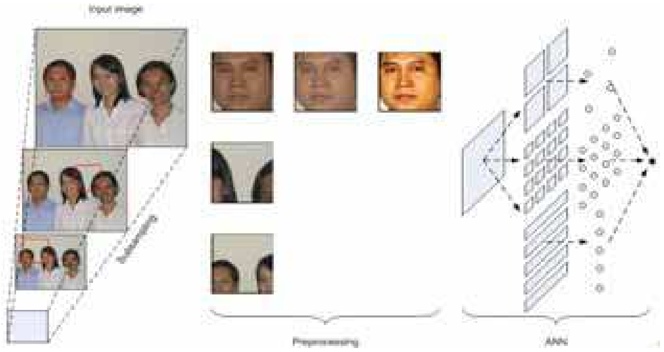
In general, human face verifying problem can be defined as following: it is the problem to determine whether a certain image contains any human face or not. Solving the problem is a two-phase process:

Phase 1. Network training phase

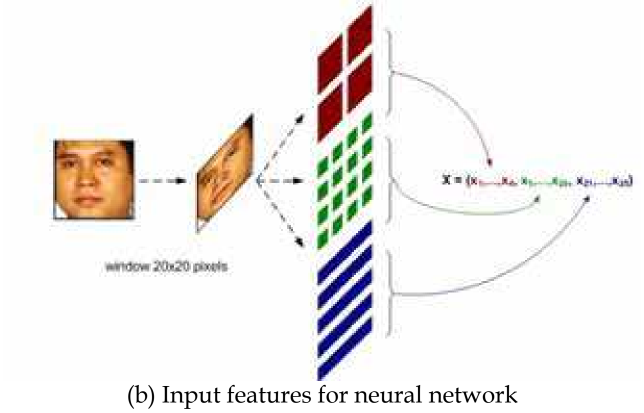
- Preparing a sample image set for training. A training sample set must contain two subsets: a subset of human face images and a subset of nonface images.
- Training the system by the sample. After the training phase, the neural network with the new set of weights will be able to verify an image.

Phase 2. Image verifying phase

- Feed the image-to-be-verified to trained neural network. The neural network will return a result: *True* if the image contains a human face or *False* if the image does not contain a human face.



(a) Processing steps of Rowley's system



(b) Input features for neural network

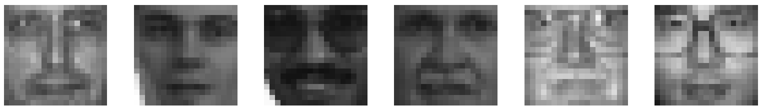
Fig. 5. Rowley's system for detecting faces

The selected neural network here is three-layer feedforward neural network with back propagation algorithm. The number of input neurons T is equivalent to the length of extracted feature vector, the number of output neurons is just 1 (C=1), 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 basing on the experiment; it depends on the sample database set of images.
One of the best face verifying system is an ANN based system developed by Rowley (H. A. Rowley et al., 1999).
The input of this first stage is a pre-processed square image (20x20 pixels) and the output of the ANN is a real value between -1 (false) and +1 (true). The preprocessing and ANN steps are illustrated in Figure 5.
The original image is decomposed into a pyramid of images as the following: 4 blocks 10x10 pixels, 16 blocks 5x5 pixels, 5 overlapping blocks 20x6 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 eyes, noses and mouths.
The system uses one hidden layer with 25 nodes to represent local features that characterize faces well. Its activation function is Tanh function with the learning rate ε = 0.3 (H. A. Rowley et al., 1999).

C. Analyses and Evaluations on AdaBoost and ANN

1. Database for experiments

Image set for training: The training database for AdaBoost and ANN has two subsets. One of them contains face images and the other contains nonface images. The database is collected from sources of CMU database and MIT database. The image set containing faces includes 2429 frontal face images. The image set not containing faces includes 4548 images which are landscape, animal images.
Image set for testing: The testing database includes images which have different size, illumination, pose and expression. These images are extracted from testing images of CalTech database. CalTech database have 450 color images.


(a) Face images

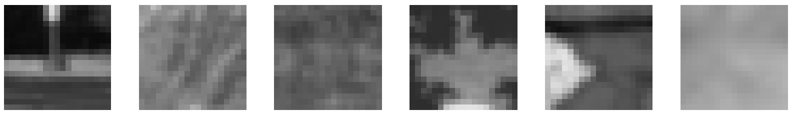

(b) Nonface images

Fig. 6. Some images for training in CMU and MIT database.

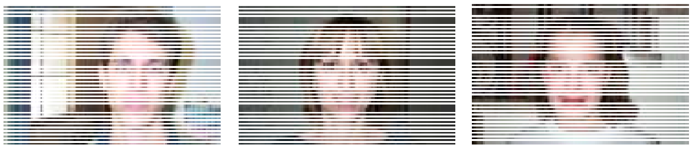


Fig. 7. Some images for testing in Caltech database.

2. Experimental results of AdaBoost based system

Database	**Number of images**	**Correct images**	**Incorrect images**		**Detecting rate**
			False negative	***False positive***	
CalTech	450	389	58	3	86,44%

Table 1. Experimental results by adaboost

The model of cascade of boosted classifiers is used, which is built with Haar-like features. This experimental model is constructed with 25 strong classifiers (n = 25) with 2888 Haar-like features and the search window size is 20 x 20 pixel. CMU and MIT database mentioned in section (2.C.1) are used to train the classifier. After that, CalTech database is used to test the trained classifier and the experimental results are presented in Table 1.

3. Experimental results of ANN based system

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

Table 2. The ann structure for detecting faces

Database	Number of images	Correct images	Incorrect images		Detecting rate
			False negative	False positive	
CalTech	450	380	25	45	84.44%

Table 3. Expermental results by the ann on caltech database

Rowley's ANN model is used for detecting faces presented in the Table II.
Thus a system is implemented by the three-layer feedforward ANN with the Tanh activation function and the back-propagation learning algorithm. The system ran on a PC, CPU 1.8MHz, RAM 512MB. CMU and MIT database mentioned in subsection (2.1.C.1) is used to train the ANN. It took about 8 hours to train the ANN. Then, CalTech database is used to test the trained ANN. The performance is presented in the Table 3.

4. Evaluations on AdaBoost and ANN

The experiments prove that AdaBoost and ANN approaches for detecting faces does 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. According to the experiments, the ratio between false negative and false positive is 58:3. To solve the drawback, there are two solutions. First, the large number of stage classifiers can be increased in order to achieve desired results. However, increasing the number of both classifiers and features too much will decrease the algorithm speed. Second, AdaBoost with other classification techniques can be combined to reject false negative images in order to increase the correctness of the system. ANN, a strong classification technique, has been used efficiently in problem of detecting faces. In addition, the performance time is not high. Since then, a hybrid model of AdaBoost and ANN is suggested in section 2.2. On the other hand, ANN is appended at the final stage to create a complete hybrid system.

2.2 A hybrid model of AdaBoost and ANN for solving the problem

1. Introduction the hybird model

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

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 8. For example, AB5 includes 5 strong classifiers will be combined with ANN, the sixth strong classifier, to be ABANN5.

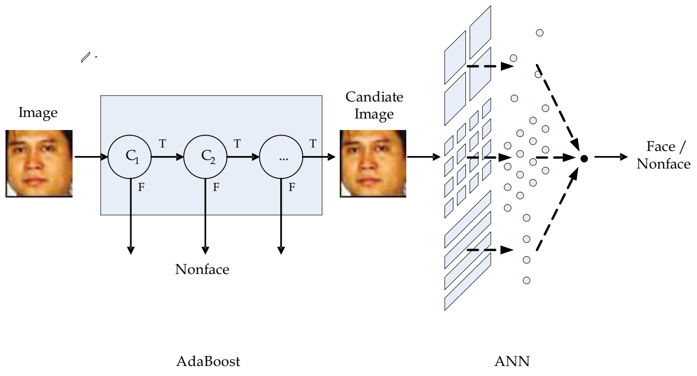


Fig. 8. The process of dectecting faces of ABANN

2. Experimental results

This exerimental also used the databases mentioned in Section (2.1.C). CMU and MIT databases were used for training; CalTech database was used for testing. Two experiments are performed. In the first experiment, AdaBoost classifiers are used with the number of stage classifiers which are 5, 10, 15, 20 or 25. The second experiment implements the AdaBoost classifiers combined with ANN which is Rowley's network with structure like three-layer feedforward neural network with 25 input nodes, 25 hidden nodes, 1 output node and using back-propagation learning algorithm for training (Tanh activation function and learning rate 0.3). These experiments were done on database CalTech. Both of them were trained by database CMU and MIT. The training time for these system is about 22 hours. The results will be showed in Table 4 and 5.

3. Evaluation of experimental results

Name	**Number of strong classifiers**	**Number of Haar-like features to use**	**correctness rate**	**Testing time (s) for CalTech**
AB5	5	131	55.86%	202
AB10	10	487	66.84%	247
AB15	15	1094	71.27%	270
AB20	20	1905	77.67%	337
AB25	25	2888	86.44%	382

Table 4. Experimental result of casscade of adaboost(AB) without ANN on caltech database (train by CMU and MIT database).

The experimental results showed that system ABs yields low diction rate even though there is a large increase in strong classifiers. For example, with 25 strong classifiers and 2888 Haar-like feature, a system achieved only detection rate 86.44% on database CalTech while system ABANNs (combining AdaBoost and ANN) achieved higher ones even in cases of the number of strong classifier to be low. For instance, ABANN5 (5 strong classifiers + ANN) achieved detection rate 84.44%; ABANN10 (10 strong classifiers + ANN) 86.52% and it got the best result 97% while increasing the number of strong classifiers to 25. The performance time is quite fast and the structure of neural network (Rowley's model) is fair simple. From the achieved results and theoretical analyses presented in Section 2.1.C, the hybrid model of associating AdaBoost with ANN is necessary and can be applied in practicality.

Name	**ADABoost structure**		**ANN structure**	**Correctness rate**	**Testing time (s) for CalTech**
	Number of strong classifiers	***Number of Haar-like features to use***			
ABANN5	5	131	Rowley's model	84.44%	247
ABANN10	10	487		86.52%	292
ABANN15	15	1094		91.30%	315
ABANN20	20	1905		94.44%	382
ABANN25	25	2888		97.15%	427

Table 5. Experimental result of system associating casscade of adaboost with ANN on caltech database (train by CMU and MIT database).

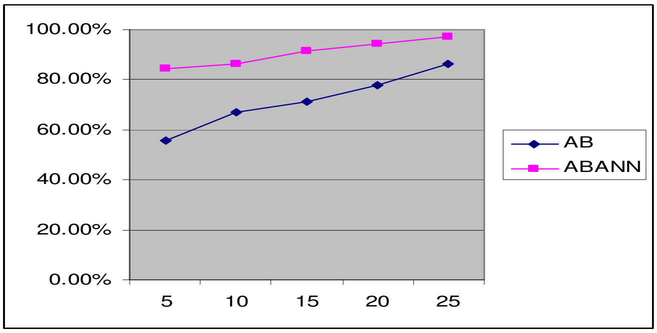


Fig. 9. Experimental resutls

2.3 Conclusions

To the extent of the paper, the two popular methods of detecting faces are presented, AdaBoost and ANN, analyzing, evaluating ones' advantages and disadvantages. From the study, 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); and 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 their limitations, the hybrid model of AdaBoost and ANN (ABANN) for detecting faces is used. The system has achieved better results of both correctness rate and performance comparing with individual models (AdaBoost or ANN). For example, ABANN25 achieved the detection result 97.15% comparing to AB25 86.44 % and ANN (based on Rowley's model) 84.44% on database CalTech and the testing time is insignificant. Since then, the hybrid model is very efficient and has a practical meaning in the problem of detecting faces.

3. Local texture classifiers based on multi layer perceptron for face alignment (Thai et al., 2008)

Face recognition is the problem to search human faces in large image database. 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. The face recognition system's stages is illustrated in Figure 1.
The face alignment is one of 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 10).

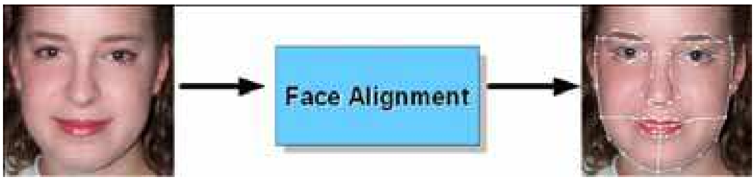


Fig. 10. Face alignment

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 et al (2001).
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 1-D profile texture model; in the second stage, updating the shape parameters which best fits these new label positions. 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 section, the classical ASM method is developed to create a new method named MLP-ASM which has achieved better results.
Because ASM only uses a 1-D 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, e.g. Gabor wavelet (Jiao et al., 2003), Haar wavelet (Zuo et al., 2004), Ranking-Boost (Yan et al, 2003) and FisherBoost (Tu et al., 2004). Nevertheless, an accurate local texture model to large data sets is still unachieved target.

In the section, an ASM method is proposed with a novel local texture model, which use Multi Layer Perceptron (MLP) for ASM local searching. MLP is very sufficient for face detecting (Marcel et al., 2004).
This section is structured as follows: in subsection 3.1, general ASM algorithm is presented; in subsection 3.2, classical local texture model with statistical local searching is discussed; in subsection 3.3, MLP local texture model is proposed for the ASM local searching; in subsection 3.4, the experimental results are presented; in section 3.5, some conclusions about the proposed model is discussed.

3.1 General ASM algorithms

A. 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 0,is important that the shapes represented are in the same coordinate frame. Figure 11 illustrates Shape Model.
In particular, a parameterized model of the form X = **Model**(b) need be found, where b is a vector of parameters of the model. Such a model can be used to generate new vectors, X. If the distribution of parameters, $p_b(b)$ can be model, we can limit them so the generated X's are similar to those in the training set. Similarly, it should be possible to estimate $p_X(X)$ using the model.

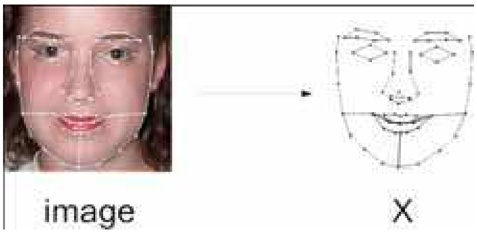


Fig. 11. Shape model of an image

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 Li et al. 2005.
Step 1 Compute the mean of the data set

$$\bar{X} = \frac{1}{s}\sum_{i=1}^{s} X_i \tag{4}$$

Step 2 Compute the covariance matrix of the data set

$$S = \frac{1}{s-1}\sum_{i=1}^{s}\left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)^T \tag{5}$$

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

$$X \approx \bar{X} + P_s b_s \tag{6}$$

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

$$b_s = P_s^T\left(X - \bar{X}\right),\, b_i \in \left\{-3\sqrt{\lambda_i}, +3\sqrt{\lambda_i}\right\} \tag{7}$$

A real shape **X** of images can be generated by applying a suitable transformation **T** to the points X: $\mathbf{X} = \mathbf{T}\left(\bar{X} + P_s b_s, x_c, y_c, s_x, s_y, \theta\right)$. This transformation includes a translation (x_c, y_c), a scaling (s_x, s_y) and a rotation (θ).

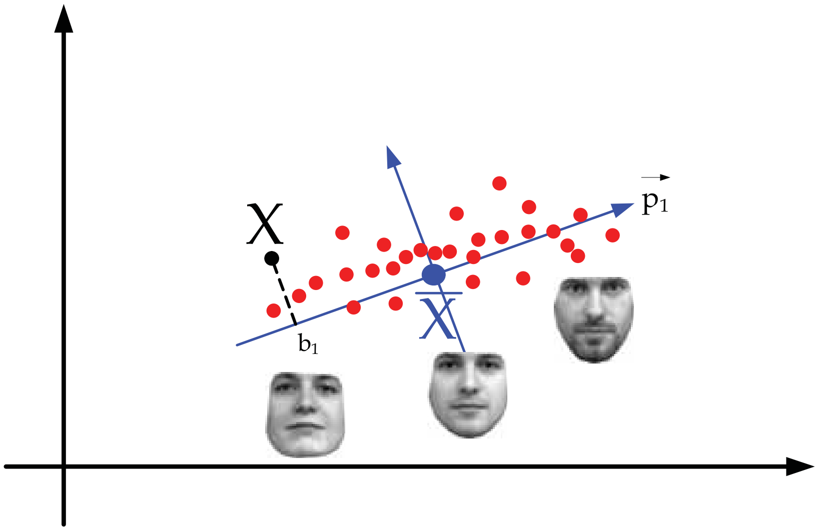


Fig. 12. Using PCA to compute statistical shape model

B. 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, $\mathbf{T}\left(\bar{X} + P_s b_s, x_c, y_c, s_x, s_y, \theta\right)$, to an image proceeds as follows.

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 3.2, then our method, the MLP local texture model will be presented in subsection 3.3.

Step 2 Repeat until convergence.

Update the parameters (b_s, x_c, y_c, s_x, s_y, θ) to best fit to the new found points X′ to minimize the sum of square distances between corresponding model and image points $E(\mathbf{b}_s, x_c, y_c, s_x, s_y, \theta) = |\mathbf{X}' - T(\bar{X} + \mathbf{P}_s\mathbf{b}_s, x_c, y_c, s_x, s_y, \theta)|^2$

Step 2.1 Fix b_s and find (x_c, y_c, s_x, s_y, θ) to minimize E

Step 2.2 Fix (x_c, y_c, s_x, s_y, θ) and find b_s to minimize E

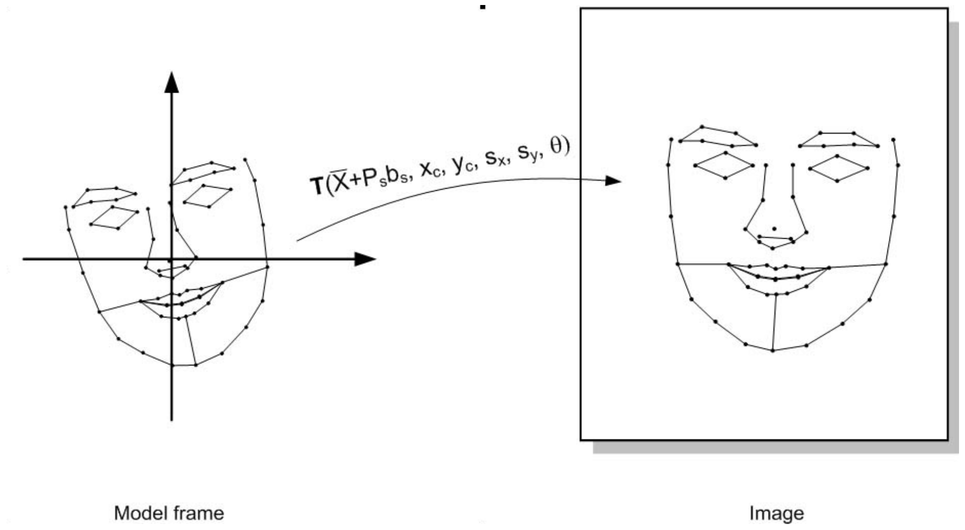


Fig. 13. Transformation model into image

2.2 Classical local texture model

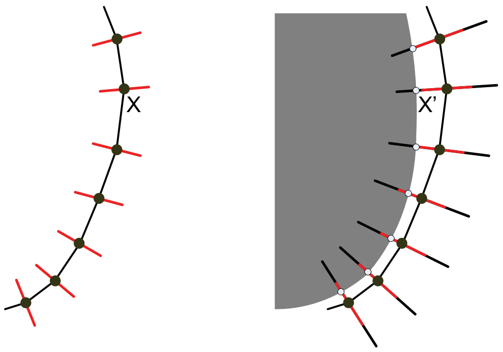


Fig. 14. 1-D profile texture model

Objective is to search for local match for each point (illustrated in figure 14.). The model is assumed to have strongest edge, correlation, statistical model of profile.

Step 1 Computing normal vector at point (x_i, y_i)

Calculating tangent vector t

$$t_x = x_{i+1} - x_{i-1},\ \ t_y = y_{i+1} - y_{i-1} \tag{8}$$

Normalize tangent vector t

$$t_x = t_x/|t|,\ \ t_y = t_y/|t| \tag{9}$$

Calculate normal vector n

$$n_x = -t_y,\ n_y = t_x \tag{10}$$

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

$$G(k) = \text{image}[x_i + kn_x, y_i + kn_y],\ k \in [\ldots, -2, -1, 0, 1, 2, \ldots] \tag{11}$$

To noise images, average orthogonal to the 1-D profile

$$g(k) = \frac{1}{4}g_{kl} + \frac{1}{2}g_{kc} + \frac{1}{4}g_{kr} \tag{12}$$

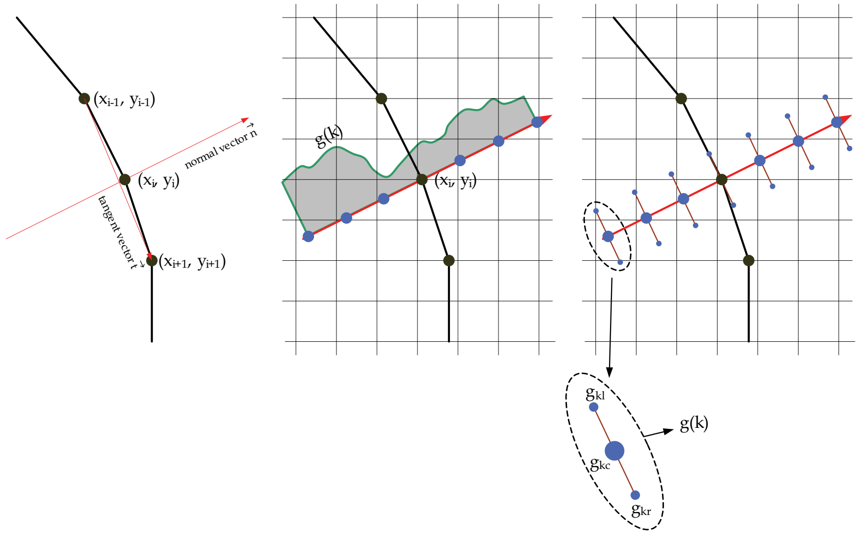


Fig. 15. Computing g(k) function

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 1-D profile from the training data set to search for the correct point.
The classical ASM method has some weak points, such as, since PCA did not consider discriminative criterions between positive samples (feature points or true points) and negative samples (non-feature points, its neighbors), the result of local searching stage often falls into local minima.
To deal with the problem, distinguishing feature points from non-feature points, which is 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. Multi Layer Perceptron has been proven to be robust and efficient in face detection (Bishop et al., 2006, Marcel et al., 2004).

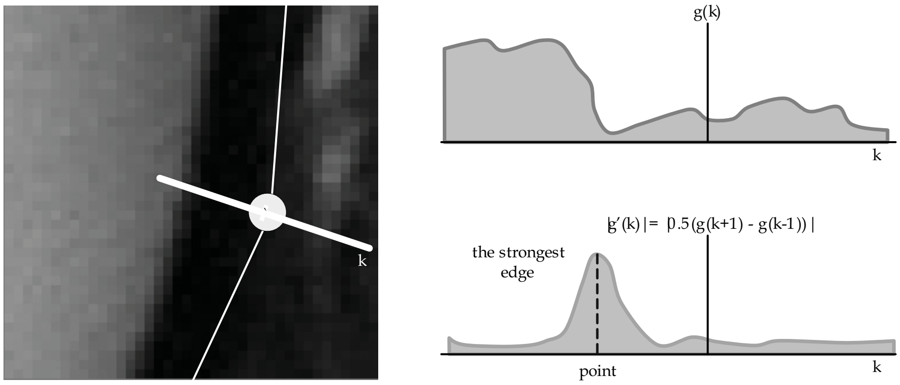


Fig. 16. Selecting the feature point at the strongest edge

3.4 Multi layer perceptron model for local texture classification

A. Structure of Multi Layer Perceptron (Bishop et al., 2006, Marcel et al., 2004)

A Multi Layer Perceptron (MLP) is a function

$$\hat{y} = MLP(x, W),\ with\ x = (x_1, x_2, \dots, x_n) \text{ and } \hat{y} = (\hat{y}_1, \hat{y}_2, \dots, \hat{y}_m) \tag{13}$$

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

For each unit i of layer L of the MLP
Integration:

$$s = \sum_j y_j^{L-1} w_{ij}^L + w_{i0}^L \tag{14}$$

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

$$\mathrm{f}(\mathrm{x}) = \begin{cases} -1 & x \le -\dfrac{1}{a} \\ ax & -\dfrac{1}{a} < x < +\dfrac{1}{a} \\ 1 & x \ge +\dfrac{1}{a} \end{cases} \tag{15}$$

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

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

The MLP uses the algorithm of Gradient Back-Propagation for training to update W.

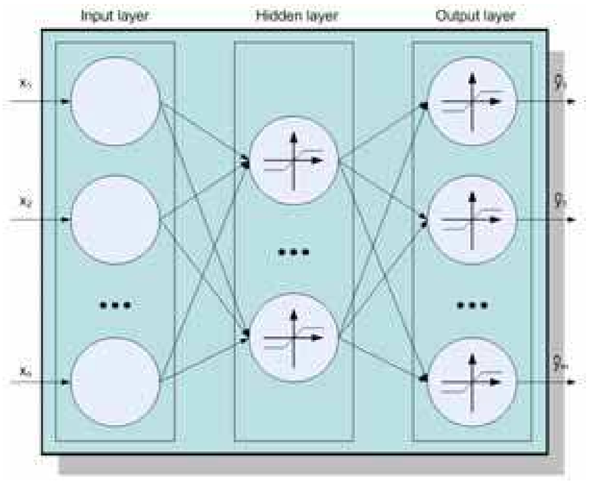


Fig. 17. Multi Layer Perceptron structure

B. Applying the Multi Layer Perceptron for Searching for Feature Points

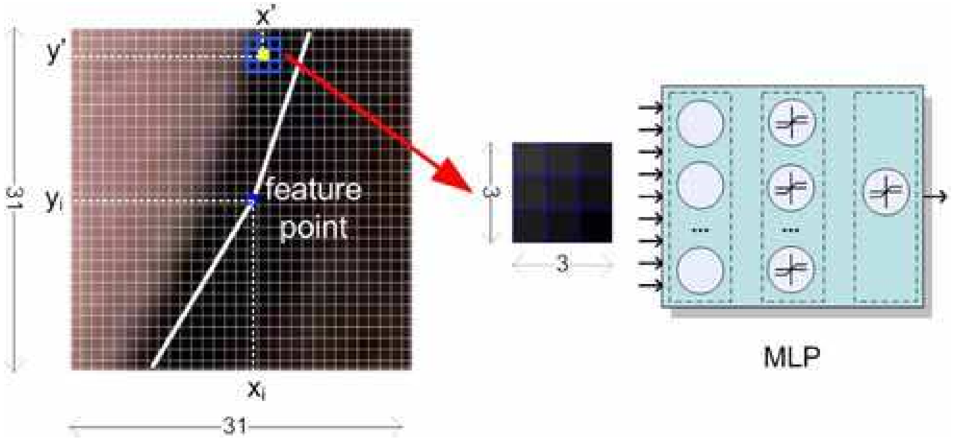


Fig. 18. Multi layer perceptron for searching for feature points

For each feature point, we define the region of a [-15, 15]×[-15, 15] window centered at the feature point. Then, positive samples, feature points, are collected from image points within a [-1, 1]×[-1, 1] sub-window at the center, while negative samples, none feature point are sampled randomly out of the sub-window within the region. Then through learning with Gradient Back-Propagation (Marcel, 2004), 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 non-feature point. Figure 18 illustrates MLP for searching for feature points.

The MLP structure for a sub-window has three layers: an input layer, a hidden layer, 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 ∈ {-1, 1}). Such that, the MLP has *(9 inputs + 1 bias) × 9 + 9 + 1 bias = 100 parameters*. The

MLP use the transfer function as a linear function with a = 0.5 (equation (12)) (this is the best fit value from the experiments over CalTech database (Weber, 1999)).
A local searching procedure will find around the current feature point to be the new feature position.

```
**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 $(x_i, y_i)$.
-    Computing MLP(sw′, W). If the return value is (+1) then point $(x', y')$ is at the edge.
-    Selecting the nearest point $(x', y')$ to the point $(x_i, y_i)$ as the new feature point $(x_i', y_i')$.
```

3.5 Experiment results

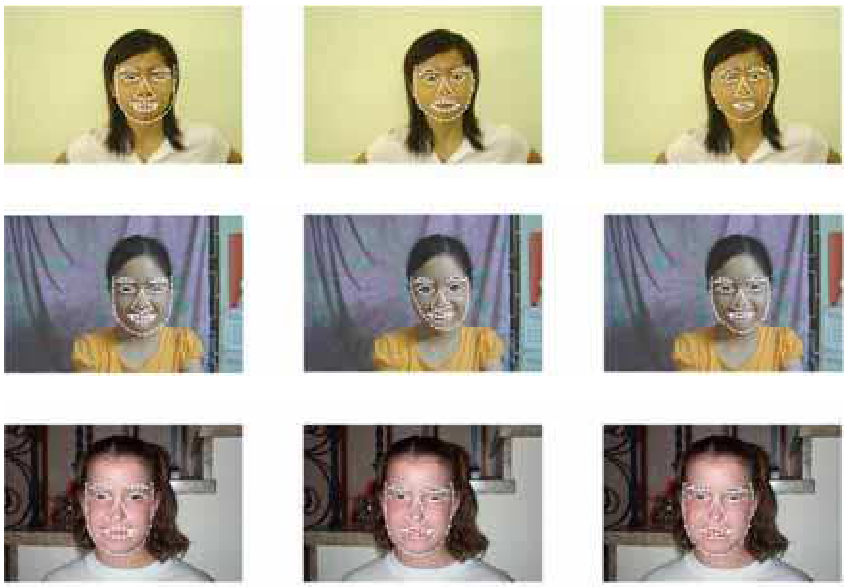


Fig. 19. Experimental results on some images from CalTech database.

We have conducted experiments on a very large data set consisting of 1,000 front face images (450 color images from CalTech data set (Weber, 1999), 550 ones from our data set). They include male and female aging from young to old people, many of which are with exaggerated expressions such as smiles, closed eyes. The average face size is about 320×320 pixels. We randomly chose 600 images for training, and the rest 400 images for testing. The face shape model is made up of 89 feature points, and for each feature point a MLP is trained.
For comparison, classical ASM was also implemented and trained on the same training set.

A. Accuracy

The accuracy is measured with point-point error. The feature points were initialized from the face window which was detected by Ada-Boost (Bradski et al., 2005, Thai et al., 2008). After the alignment procedure, the errors were measured. The average errors of the 89 feature points are compared in Figure 20. The x-axis, which represents the index of feature points, is grouped by organ. It shows that the proposed method outperforms classical ASM; especially, the improvement of the methods is mainly on feature points of mouth and contour.

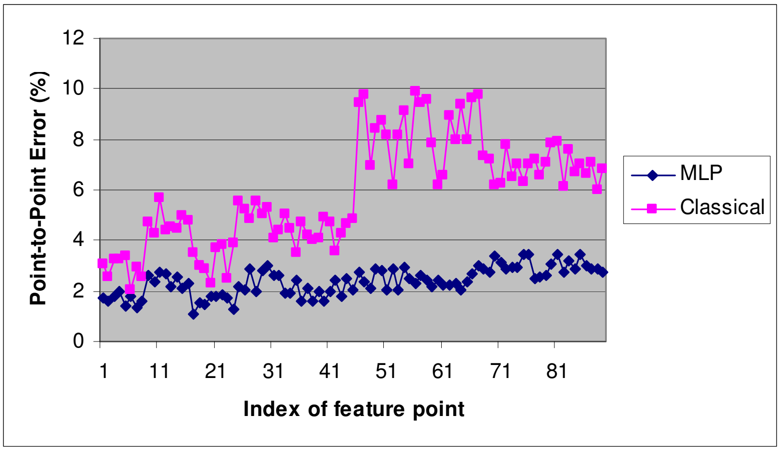


Fig. 20. Comparison of classical ASM and MLP ASM

B. Efficiency

The average performance time is listed in Table 6. All the tests are carried out on a PC with Intel Pentium IV - 2.4G CPU, 1G RAM. The classical ASM is the fastest since its computation of local texture model is very simple. The method is a little slower but is still comparable with the classical ASM.

Algorithm	**Classical ASM**	**MLP ASM**
Time per iteration	2ms	56ms

Table 6. The average performance time per iteration (a two-stage process).

3.6 Conclusions

In the section 3, we present a robust face alignment algorithm with a local texture model (MLP ASM). Instead of modeling a local feature by 1-D profile texture, the classifier is learned from its 2-D profile texture patterns against its neighbor ones as a local texture model. The classifier is of 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.

4. Facial expression classification based on mutil Artificial Neural Network (Thai et al., 2010)

There are many approaches apply for image classification. At the moment, the popular solution for this problem: using K-NN and K-Mean with the different measures, Support Vector Machine (SVM) and Artificial Neural Network (ANN).

K-NN and K-Mean method is very suitable for classification problems, which have small pattern representation space. However, in large pattern representation space, the calculating cost is high.
SVM method applies for pattern classification even with large representation space. In this approach, we need to define the hyper-plane for classification pattern [1]. For example, if we need to classify the pattern into L classes, SVM methods will need to specify 1+ 2+ … + (L-1) = L (L-1) / 2 hyper-plane. Thus, the number of hyper-planes will rate with the number of classification classes. This leads to: the time to create the hyper-plane high in case there are several classes (costs calculation). Besides, in the situation the patterns do not belong to any in the L given classes, SVM methods are not defined (Brown et al., 2005). On the other hand, SVM will classify the pattern in a given class based on the calculation parameters. This is a wrong result classification.
One other approach is popular at present is to use Artificial Neural Network for the pattern classification. Artificial Neural Network will be trained with the patterns to find the weight collection for the classification process (Kiem et al., 2000). This approach overcomes the disadvantage of SVM of using suitable threshold in the classification for outside pattern. If the patterns do not belong any in L given classes, the Artificial Neural Network identify and report results to the outside given classes.
In this section, the Multi Artificial Neural Network (MANN) model is used 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 Sub Neural Network (SNN) of MANN. Lastly, MANN's global frame (GF) consisting some Component Neural Network (CNN) is used to compose the classified result of all SNN.

4.1 Multi Artificial Neural Network

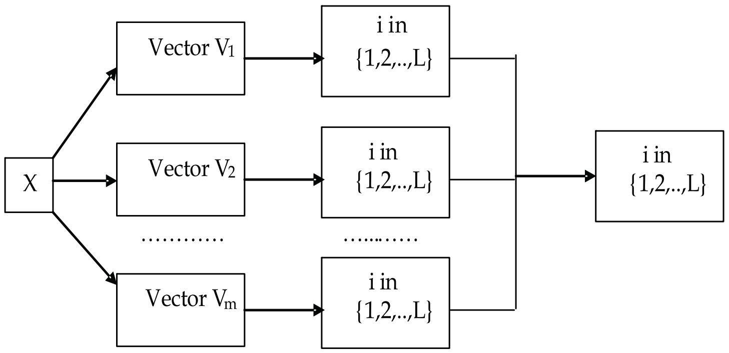


Fig. 21. Image classification

There are a lot of approaches to classify the image featured by m vectors X= $(v_1, v_2, ..., v_m)$. Each of patterns is needed to classify in one of L classes: $\Omega = \{\Omega i \mid 1 \le i \le L\}$. This is a general image classification problem (Kiem et al., 2000) with parameters (m, L).
A Sub-Neural Network will classify the pattern based on the responsive feature. To compose the classified result, we can use the selection method, average combination method or build the reliability coefficients…

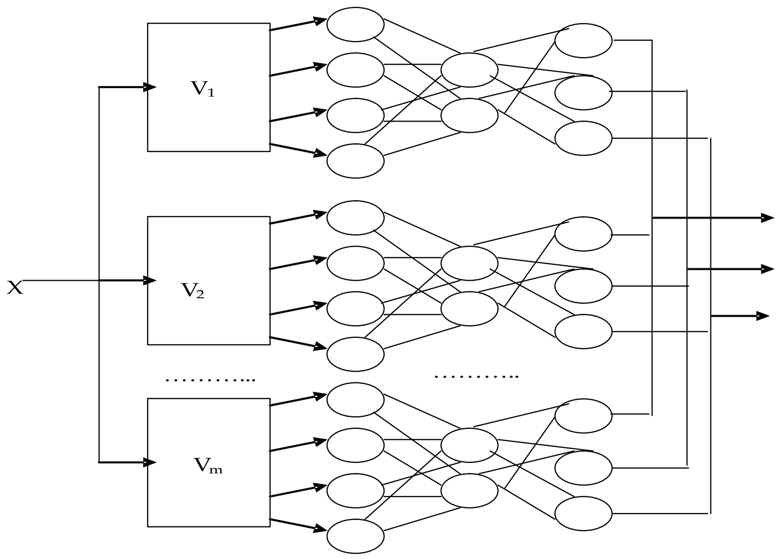


Fig. 22. Processing of Sub neural networks

The selection method will choose only one of the classified results of a SNN to be the whole system's final conclusion:

$$P(\Omega_i \mid X) = P_k(\Omega_i \mid X)\ (k=1..m) \tag{16}$$

Where, $P_k(\Omega_i \mid X)$ is the image X's classified result in the Ω_i class based on a Sub Neural Network, $P(\Omega_i \mid X)$ is the pattern X's final classified result in the Ω_i. Clearly, this method is subjectivity and omitted information.

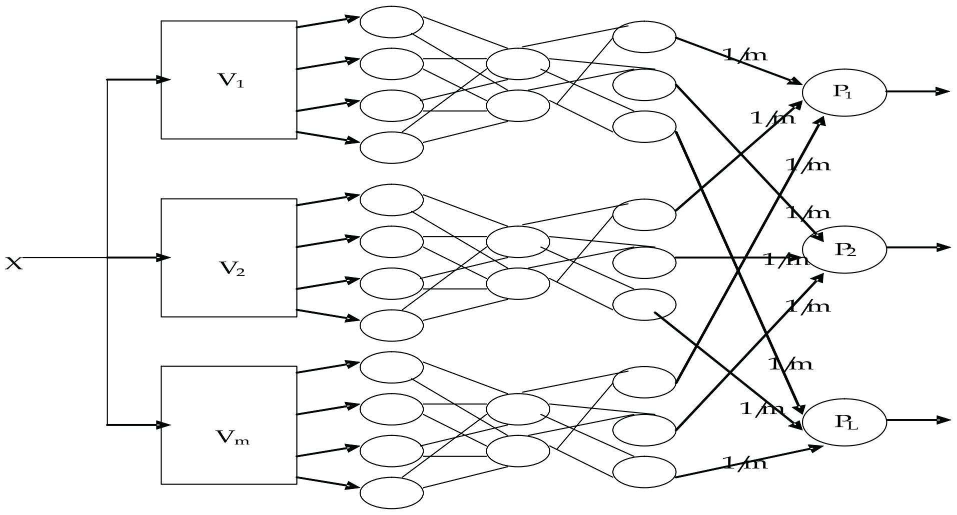


Fig. 23. Average combination method.

The average combination method (Thai et al. 2004) uses the average function for all the classified result of all SNN:

$$P(\Omega_i \mid X) = \sum_{k=1}^{m} \frac{1}{m} P_k(\Omega_i \mid X) \tag{17}$$

This method is not subjectivity but it set equal the importance of all image features.
On the other approach is building the reliability coefficients attached on each SNN's output (Thai, 2004, Bac et al., 2005). The fuzzy logic, SVM, Hidden Markup Model (HMM) (Ghoshal et al., 2005) is used to build these coefficients:

$$P(\Omega_i \mid X) = \sum_{k=1}^{m} r_k P_k(\Omega_i \mid X) \tag{18}$$

Where, r_k is the reliability coefficient of the kth Sub Neural Network. For example, the following model uses Genetics algorithm to create these reliability coefficients.

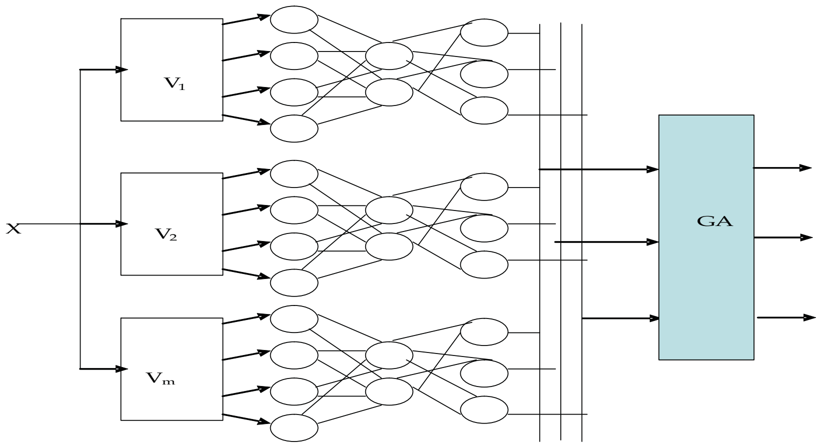


Fig. 24. NN_GA model (Thai, 2004)

In the next section, the Neural Network technique is used. In details, a global frame consisting of some CNN(s) is used. The weights of CNN(s) evaluate the importance of SNN(s) like the reliability coefficients. This model links many Neural Networks together, so it is called Multi Artificial Neural Network (MANN).

4.2 Multi Artificial Neural Network apply for image classification

A. The MANN model

Multi Artificial Neural Network (MANN), applying for pattern or image classification with parameters (m, L), has m Sub-Neural Network (SNN) and a global frame (GF) consisting L Component Neural Network (CNN). In particular, m is the number of feature vectors of image and L is the number of classes.
Definition 1: SNN is a 3 layers (input, hidden, output) Neural Network. The number input nodes of SNN depend on the dimensions of feature vector. SNN has L (the number classes)output nodes. The number of hidden node is experimentally determined. There are

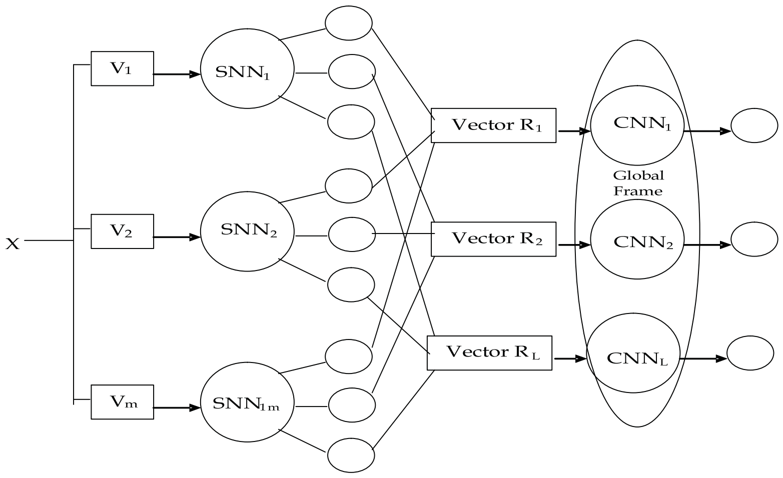


Fig. 25. MANN with parameters (m, L).

m (the number of feature vectors) SNN(s) in MANN model. The input of the ith SNN, symbol is SNN_i, is the feature vector of an image. The output of SNN_i is the classified result based on the ith feature vector of image.

Definition 2: Global frame is frame consisting L Component Neural Network which compose the output of SNN(s).

Definition 3: Collective vector kth, symbol R_k (k=1..L), is a vector joining the kth output of all SNN. The dimension of collective vector is m (the number of SNN).

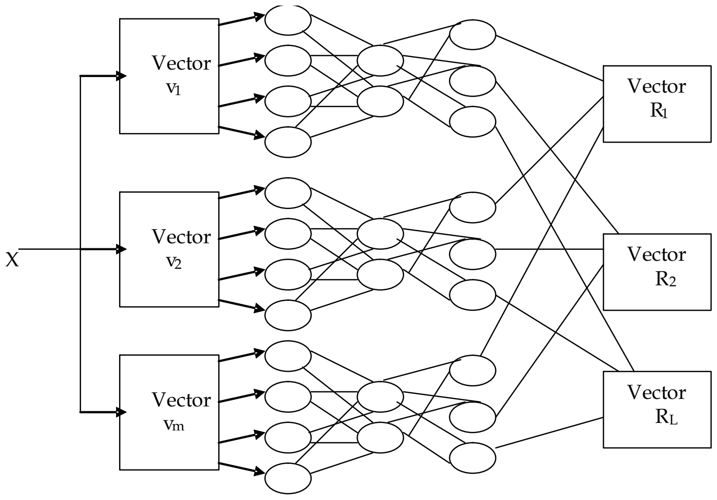


Fig. 26. Create collective vector for CNN(s).

Definition 4: CNN is a 3 layers (input, hidden, 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 node is experimentally determined. There are L CNN(s). The output of the jth CNN, symbols is CNN_j, give the probability of X in the jth class.

B. The process of the MANN model

The training process of MANN is separated in 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, the SNN_1 will be trained first. After that $SNN_2,\ldots,SNN_m$ will be trained.

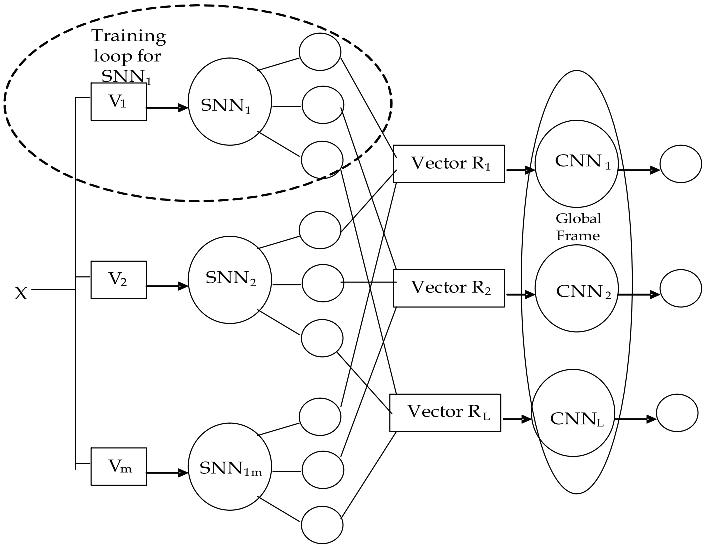


Fig. 27. SNN1 local training

In the global training phase, CNN_1 will be trained. After that $CNN_2,\ldots,CNN_L$ will be trained.

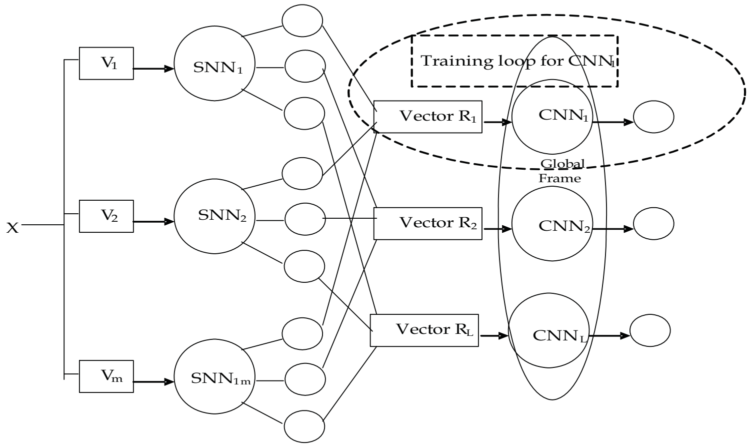


Fig. 28. CNN1 global training.

The classification process of pattern X using MANN is below: firstly, pattern X are extract to m feature vectors. The ith feature vector is the input of SNN_i classifying pattern. Join all the kth output of all SNN to create the kth (k=1..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.

C. Applying the MANN for six basic facial expressions classification

In the above section, the MANN is explained in the general case with parameters (m, L) apply for pattern classification. Now, MANN model is applied for scenery image of regional tourism classification. In fact that this is an experimental setup with (m=4, L=6). The dimensions of input vector of all SNN are not the same. An automatic facial feature extraction system is used, which is able to identify the eye location, the detailed shape of eyes and mouth, chin and inner boundary from facial images (Hung, 2009).

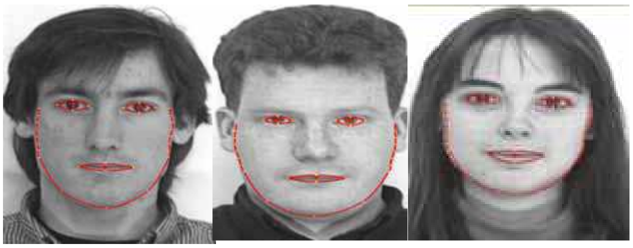


Fig. 29. All Features Extraction (Hung, 2009).

The left eye is the input for SNN_1. The right eye is the input for SNN2. When emotional expression on the face, the left eye and the right eye may not be completely matched each other..The mouth is the input for SNN_3. The inner boundary is the input for SNN_4. All SNN(s) are 6 output nodes matching to 6 basic facial expression (happiness, sadness, surprise, anger, disgust, fear) (Michael et al., 1999). The experimental MANN has 6 CNN(s). They give the probability of the face in six basic facial expressions. It is easy to see that to build MANN model only use Neural Network technology to develop the experimental system.
The proposed model is applied for 6 basic facial expressions on JAFFE database consisting 213 images posed by 10 Japanese female models. The experimental result is presented below:

Comparison	SNN1	SNN2	SNN3	SNN4	Average	MANN
Precision	71%	73%	76%	56%	80%	83%

Table 7. Facial Expression Precision

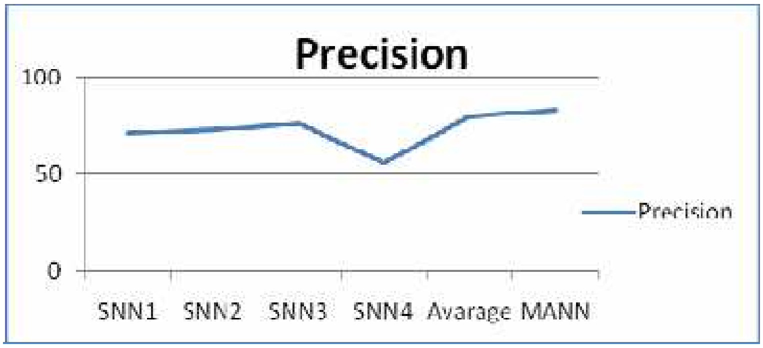


Fig. 30. Facial expression using different methods.

It is a small experimental to check MANN model and need to improve the experimental system. Although the result classification is not high, the improvement of combination result shows that the MANN's feasibility such a new method combines. We need to integrate with another facial feature sequences extraction system to increase the classification precision.

4.3 Conclusion

In this section, the model Multi Artificial Neural Network (MANN) with parameters (m, L) is explained. This model is applied for facial expression or image classification. Include, m is the number of feature vectors of image. L is the number of classes. MANN model has m Sub-Neural Network SNN_i (i=1..m) and a Global Frame (GF) consisting L Components Neural Network CNN_j (j=1..L). Each of SNN uses to process the responsive feature vector. Each of CNN use to combine the responsive element of SNN's output vector. In fact, the weight coefficients in CNN_j are as the reliability coefficients the SNN(s)' the j^{th} output. It means that the importance of the ever feature vector is determined after the training process. On the other hand, it depends on the image database and the desired classification.
To experience the feasibility of MANN model, in this study, we conducted to develop a MANN model with parameters (m=4, L=3) apply for six basic facial expressions on JAFFE database. The experimental result shows that the proposed model improves the classified result compared with the selection and average combination method.

5. Conclusion

This chapter presented some methods for the recognition of human faces using many different modes of artificial neural network and combination models. The proposed techniques are based on the AdaBoost and Artificial Neural Network (AANN) structure, Multi Layer Perceptron (MLP) structure and Multi Artificial Neural Nework (MANN) structure. An implementation example is given to demonstrate the feasibility of each technique for the human face recognition system. The higher recognition rate conventional techniques on the standard database was obtained using these proposed techniques.These show the feasibility of many modes of articial neural network for facial image proceesing.

6. References

C. Bishop (2006), Pattern Recognition and Machine Learning, Springer Press.

H.Schneiderman (2000), A Statistical Approach to 3D Object Detection Applied to Faces and Cars, *CMU*.

S.Z. Li, and A.K. Jain (2005), Handbook of face recognition, Springer Press.

S.Z. Li, and Z. Zhang (2004), FloatBoost learning and statistical face detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1112-1113.

H.A. Rowley (1999), Neural network Based Face Detection, *Book Neural network Based Face Detection, Series Neural network Based Face Detection*, School of Computer Science, Computer Science Department, Carnegie Mellon University Pittsburgh, PA 15123, 1999

M.A. Turk, and A.P. Pentland (1991), Eigenface for recognition, *Journal of Cognitive Neuroscience*, pp. 76-86.

P. Viola, and M. Jones (2001), Rapid object detection using a boosted cascade of simple features, Proc. Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 511-518.

P. Viola, and M. Jones (2004), Robust real-time face detection, *International Journal of Computer Vision*, pp. 137-154.

M.H. Yang, D.J. Kriegman, and N. Ahuja (2002), Detecting Faces in Images: A Survey, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, pp. 34-47.

Tong, S. and E. Chang (2001), Support vector machine active learning for image retrieval, *Proceedings of the ninth ACM international conference on Multimedia*, p. 107-118.

Brown, R. and B. Pham (2005), Image Mining and Retrieval Using Hierarchical Support Vector Machines, *Proceedings of the 11th International Multimedia Modeling Conference (MMM'05*, p. 446-451.

Hoang Kiem, Le Hoai Bac, Le Hoang Thai (2000), Neural Network and Genetic Algorithm apply for finger recognizes, *the second conference: Informatics Technology Department, Natural Science University*, HCM City, Vietnam.

Le Hoang Thai (2004), Building, Development and Application Some Combination Models of Neural Network (NN), Fuzzy Logic (FL) and Genetics Algorithm (GA), PhD Mathematics Thesis, Natural Science University, HCM City, Vietnam.

Le Hoai Bac, Le Hoang Thai (2004), the GA_NN_FL associated model for authenticating finger printer, *the KES'2004 International Program Committee, Wellington Institute of Technology*, NEW ZEALAND, pp. 708-715.

Ghoshal, A., P. Ircing, and S. Khudanpur (2005), Hidden Markov models for automatic annotation and content-based retrieval of images and video, *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, p. 544-551.

Chen, Y. and J.Z. Wang (2002), A region-based fuzzy feature matching approach to content-based image retrieval, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, pp. 1252-1267.

Hoiem, D. et al. (2004), Object-based image retrieval using the statistical structure of images, *Proceedings of the IEEE Computer Society Conference on CVPR 2004*, pp. 490-497 .

Siu-Yeng Cho and Zheru Chi (2005), Genetic Evolution Processing of Data Structure for Image Classification, *IEEE Transaction on Knowledge and Data Engineering*, Vol 17, No 2, pp. 216-231.

Nguyen Viet Hung (2009), Facial Expression Based on Wavelet Transform, *the 2nd International Congress on Image and Signal Processing* (CISP'09), pp. 330-339.

Michael J. Lyons, Julien Budynek, & Shigeru Akamatsu (1999), Automatic Classification of Single Facial Images, *IEEE Transactions on Pattern Analysis and Machine Intelligence 21*, pp. 1357-1362.

Thai Hoang Le, Len Tien Bui (2008), An approach to combine adaboost and artificial neural network for detecting human faces, proceeding of the 2008 IEEE International Conference on System, Man, Cybernetics (SMC 2008), pp. 3411-3415.

Thai Hoang Le, Len Tien Bui (2008), Local texture classifiers based on multi layer perceptron for face alignment, ICTFIT'08, pp. 32-38.

Thai Hoang Le, Hai Son Tran (2010), Facial expression classification based on mutil artificial neural network, proceedings of International Conference on Advanced Computing and Applications (ACOMP), HCM City University of Technology, pp. 125 – 133.