Available online at www.sciencedirect.com
SCIENCE DIRECT®
Pattern Recognition Letters 27 (2006) 1289–1298
Pattern Recognition Letters
www.elsevier.com/locate/patrec

Evolutionary feature synthesis for facial expression recognition

Jiangang Yu *, Bir Bhanu

Center for Research in Intelligent Systems, University of California, Riverside, CA 92521, USA

Available online 10 January 2006

Abstract

Feature extraction is one of the key steps in object recognition. In this paper we propose a novel genetically inspired learning method for facial expression recognition (FER). Unlike current research on facial expression recognition that generally selects visually meaningful feature by hands, our learning method can discover the features automatically in a genetic programming-based approach that uses Gabor wavelet representation for primitive features and linear/nonlinear operators to synthesize new features. These new features are used to train a support vector machine classifier that is used for recognizing the facial expressions. The learned operator and classifier are used on unseen testing images. To make use of random nature of a genetic program, we design a multi-agent scheme to boost the performance. We compare the performance of our approach with several approaches in the literature and show that our approach can perform the task of facial expression recognition effectively.

Keywords: Genetic programming; Feature learning; Gabor filters

1. Introduction

Automatic face expression recognition (FER) is desirable for a variety of applications such as human–computer interaction, human behavior understanding, perceptual user interface, and interactive computer games. In an automatic FER system, face detection or localization in a cluttered scene is usually the first step. Next, relevant features from the face must be extracted, and finally the expression can be classified based on the extracted features (Daugman, 1997; Pantic and Rothkrantz, 2000).

As compared to face recognition, there is a relatively small amount of research on facial expression recognition. Previous work on automatic facial expression includes studies using representations based on optical flow, principal components analysis and physically based models. Viola uses Adaboost method to solve computer vision problems such as image retrieval and face detection (Viola and Jones, 2001), which can select features in the learning phase using a greedy strategy. AdaBoost method does not perform well in the small sample case (Guo and Dyer, 2003), which is used in our experiments. Yacoob and Davis (1994) use the inter-frame motion of edges extracted in the area of the mouth, nose, eyes, and eyebrows. Bartlett et al. (1996) use the combination of optical flow and principal components obtained from image differences. Hoey and Little (2000) approximate the flow of each frame with a low dimensional vector based upon a set of orthogonal Zernike polynomials and apply their method to the recognition of facial expressions with hidden Markov models (HMMs). Lyons et al. (1998, 1999), Zhang et al. (1998) and Zhang (1999) use Gabor wavelet coefficients to code face expressions. In their work, they first extract a set of geometric facial points and then use multi-scale and multi-orientation Gabor wavelets filters to extract the Gabor wavelet coefficients at the chosen facial points. Similarly, Wiskott et al. (1997) use a labeled graph, based on the Gabor wavelet transform, to represent facial expression images. They perform face recognition through elastic graph matching.

* Corresponding author.
E-mail addresses: jyu@cris.ucr.edu (J. Yu), bhanu@cris.ucr.edu (B. Bhanu).

0167-8655/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2005.07.026

Facial feature extraction attempts to find the most appropriate representation of face images for recognition and it is the key step in facial expression recognition. The extracted features capture the characteristics of face expressions and are fed to a classifier for recognition. The recognition accuracy of an automatic facial expression recognition system is determined by the quality of the feature set used. What are the good features? How can we synthesize effective features automatically based on the available information? It is difficult to identify a set of features that characterize a complex set of facial expressions. Typically, many types of features are explored before a recognition system can be built to perform the desired recognition task. There are a lot of features available and these features may be correlated, making the design and selection of appropriate features a very time consuming and expensive process.

For conventional methods, human experts design an approach to detect potential features in images depending on their knowledge and experience. This approach can often be dissected into some primitive operations on the original image or a set of related feature images obtained from the original one. Human experts try only some limited number of conventional combinations and explore a very small portion of the feature space since they are biased with their knowledge and have limited computational capability. On the other hand, GP, however, may try many unconventional ways of combining primitive operations that may never be imagined by a human expert. Although some of these unconventional combinations could be difficult to be explained by human experts, in some cases, it is these unconventional combinations that yield exceptionally good recognition results. In addition, the inherent parallelism of GP and the high speed of current computers allow the portion of the search space explored by GP to be much larger than that by human experts, enhancing the probability of finding an effective composite operator. The search performed by GP is not a random search. It is guided by the fitness of composite operators in the population. As the search proceeds, GP gradually shifts the population to the portion of the feature space containing good composite operators. Tan et al. (2003) propose a learning algorithm for fingerprint classification based on GP. Bhanu and Yu use GP for facial expression recognition with a Bayesian classifier (Bhanu et al., 2004). Unlike the conventional methods that select visually meaningful features by hand (Lyons et al., 1998, 1999; Zhang et al., 1998; Zhang, 1999; Guo and Dyer, 2003), our approach can synthesize the features automatically. For the features chosen by hand, the points which are chosen are highly dependent on the person and the database. Our proposed approach learns features without resorting to a specific database. Therefore, our approach could be considered as fully domain-independent. To the best of our knowledge, unconventional features discovered by the computer have never been used in facial expression classification.

Section 2 presents the recognition system and explains the technical details. Experiments and results are presented in Section 3, where we compare our results with the other published work. Finally, Section 4 provides the conclusions of this paper.

2. Technical approach

Genetic programming (GP) is an evolutionary computational paradigm (Koza, 1994; Bhanu et al., 2005) that is an extension of genetic algorithm and works with a population of individuals. An individual in a population can be any complicated data structure such as linked lists, trees, graphs, etc. In this paper, individuals are composite operators represented by binary tree with primitive operators as internal nodes and primitive features as leaf nodes. We design different primitive operators, which form primitive operators pool. During the training, GP chooses primitive operators from the primitive operators pool and runs on primitive features generated from the raw facial expression images to generate composite operators, which generate the elements of composite feature vectors by combining the primitive operators. It is a way of combining primitive features.

The advantage of using a tree structure is that it is powerful enough in expressing the ways of combining primitive features and unlike a graph, it has no loops and this guarantees that the execution of individuals represented by trees will terminate and not be trapped in an infinite loop. Feature vectors are generated by the learned composite operators, which are used for facial expression recognition. The search space is the set of all possible composite operators. The primitive features can be simple features directly extracted or complicated features designed by the human experts based on the characteristics of objects to be recognized in a particular kind of imagery (e.g., facial expression images). The primitive features are Gabor filtered images in this paper. With each individual evolved by a population of GP, a composite operator is evolved. By applying composite operators to the primitive features filtered from images, composite feature vectors are obtained. These composite feature vectors are fed into a classifier for recognition. Fig. 1 shows the block diagram in our approach. This system has training and testing modules, which are shown in Fig. 2. During the training step, extracted features (Gabor filtered images) are fed into GP to evolve composite operators (binary trees), which generate composite vectors. And then composite vectors are passed to train SVM classifier. The primitive operators and primitive features are decoupled from the GP mechanism that generates composite feature vectors, so they can be tailored to a particular recognition task without affecting the other parts of the system. Thus, the method and the recognition system are flexible and can be applied to a variety of images.

During testing, the learned best composite operator is applied directly to generate feature vectors. Since the parameters of SVM classifier are determined by the feature

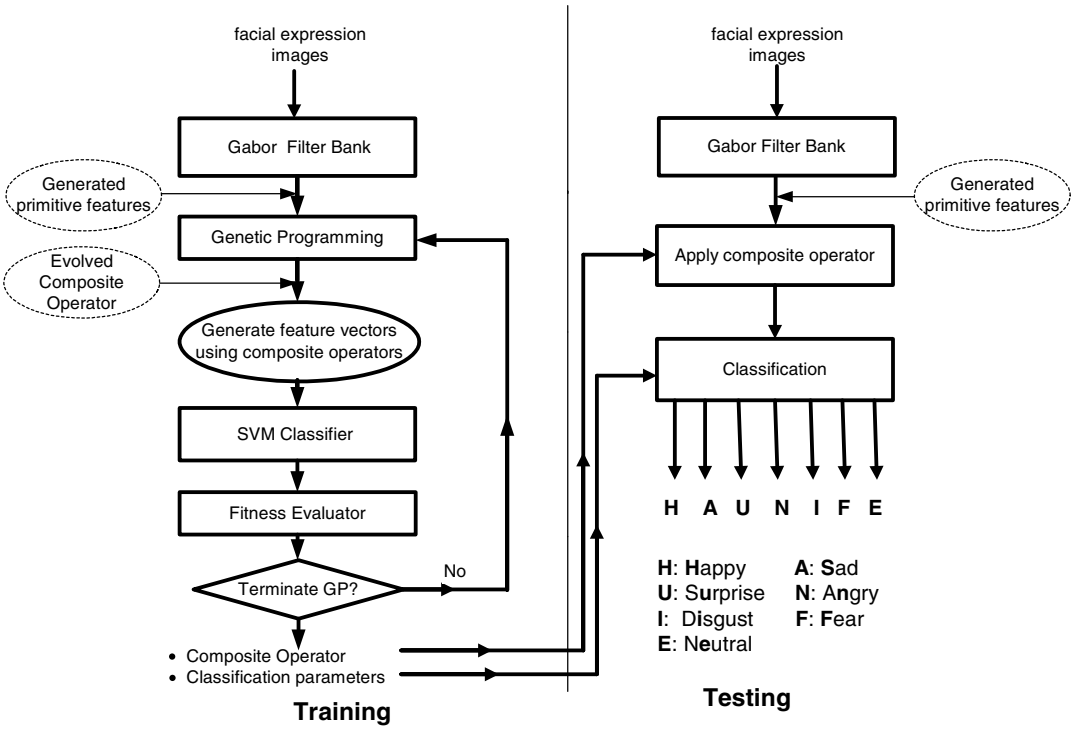

Fig. 1. Block diagram of our approach for facial expression recognition.

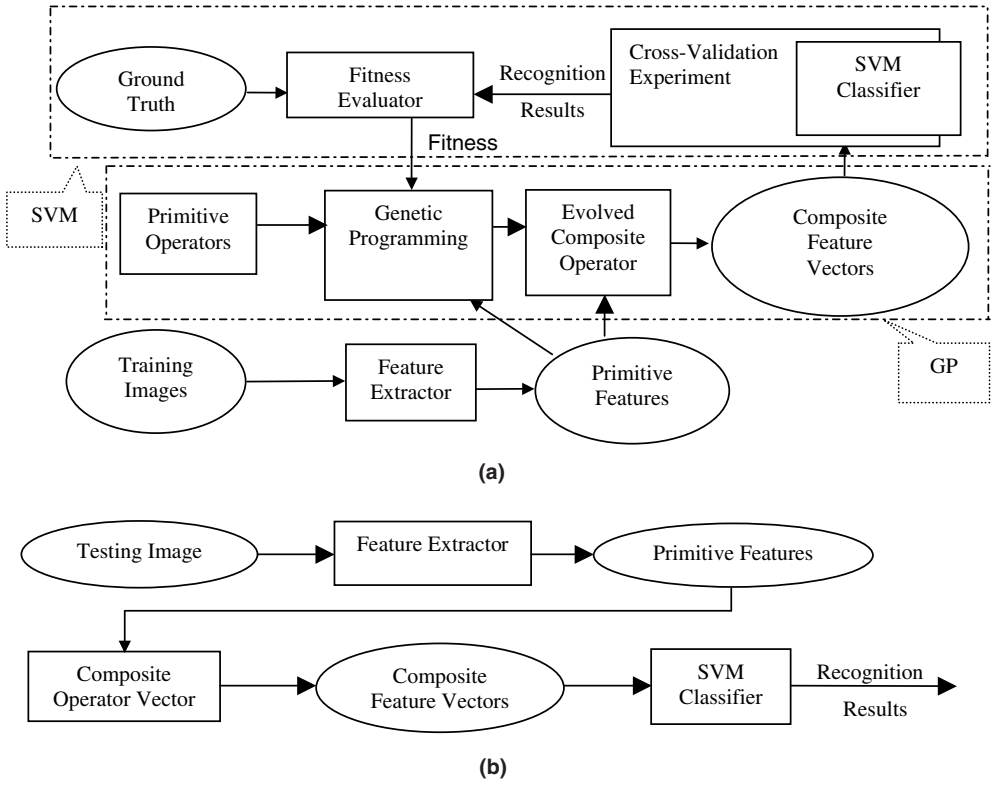

Fig. 2. Training and testing modules in our approach. (a) Training module: Learning composite operator vectors and SVM classifier. (b) Testing module: Applying learned composite operator vector and SVM classifier to a testing image.

vectors from the training, the classifier as well the composite operators are learned by using GP. Note that, in our approach, we do not need to find any reference point on the image.

2.1. Gabor filter bank

In our approach, the Gabor filtered images are used as primitive features. The Gabor filters can be considered as orientation and scale tunable edge and line detectors, which have been shown to be optimal in the sense of minimizing the joint two-dimensional uncertainty in space and frequency. The general form of a 2-D Gabor function is given as

$$g(x,y) = \frac{1}{2\pi\sigma_x\sigma_y}\exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right)+2\pi \mathrm{j} W x\right] \quad (1)$$

where (x,y) is the spatial centroid of the elliptical Gaussian window.

From Eq. (1), we can get its Fourier transform $G(u,v)$ as

$$G(\mu,v) = \exp\left\{-\frac{1}{2}\left[\frac{(\mu-W)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right]\right\} \quad (2)$$

where W is the frequency of a sinusoidal plane wave along the x-axis, and σ_x, σ_y are the space constants of the Gaussian envelop along the x- and y-axes, respectively. u, v are the frequency components in x- and y-direction, respectively, $\sigma_u = 1/2\pi\sigma_x$ and $\sigma_v = 1/2\pi\sigma_y$. Gabor function forms a complete but nonorthogonal basis set. Expanding a signal using this basis provides a localized frequency description.

Gabor filter bank with multi-orientation can be obtained by a rigid rotation of $g(x,y)$ through the generating function:

$$g(x,y) = a^{-m}G(x',y'), \quad a>1 \quad (3)$$

where

$$x' = a^{-m}(x\cos\theta + y\sin\theta) \quad \text{and}$$
$$y' = a^{-m}(-x\sin\theta + y\cos\theta) \quad (4)$$

and $\theta = n\pi/K$, θ is the rotation angle and K is the total number of orientations.

Because the area of the energy distribution of the filters varies with scale, the Gaussian envelope should vary with the filter size. We design the Gabor filter bank based on the filters used previously for texture segmentation and image retrieval (Manjunath and Ma, 1996; Jain and Farokhnia, 1991; Wu and Bhanu, 1997).

We designed the Gabor filter bank with the following parameters:

$$a = (U_h/U_l)^{1/S-1}, \quad \sigma_u = \frac{(a-1)U_h}{(a+1)\sqrt{2\ln 2}} \quad (5)$$

$$\sigma_v = \tan(\pi/2K)\left[W-\frac{(2\ln 2)\sigma_u^2}{W}\right]\left[2\ln 2-\frac{(2\ln 2)^2\sigma_u^2}{W^2}\right]^{-\frac{1}{2}} \quad (6)$$

where $W = a^m U_l$ and $m = 0,1,2,\ldots,S-1$. We define W with the scale factor a^m to ensure the energy is independent of m. U_h, U_l denote the lower and upper center frequencies of interest, respectively. $n = 0,1,2,\ldots,K-1$. m and n are the indices of scale and orientation, respectively. K is the number of orientations and S is the number of scales. In order to eliminate sensitivity of the filter response to absolute intensity values, the real components of the 2D Gabor filters are biased by adding a constant to make them zero mean. This can be easily done by making $G(0,0) = 0$. The design strategy is to ensure that the half-peak magnitude support of the filter responses in the frequency spectrum touch each other as shown in Fig. 3.

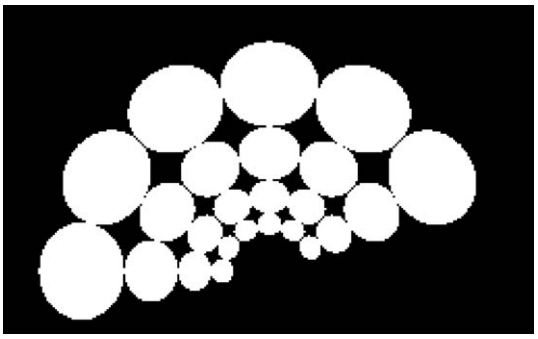

Fig. 3. The filter set in the frequency domain indicates the half-peak magnitude.

2.2. Design considerations

The set of terminals: The set of terminals used in this paper are called primitive features which are generated from the raw facial expression images filtered by Gabor filter bank at four scales and six orientations. These 24 images are input to composite operators. For simplicity, we resize the filtered image size 256×256 to 32×32. GP determines which operators are applied on primitive features and how to combine the primitive operators. Fig. 4 shows an example of primitive features obtained by Gabor filter bank (four scales and six orientations).

The set of primitive operators: A primitive operator takes one or two input images and performs a primitive operation on them and outputs a resultant image and/or feature vectors. In our approach, we designed two kinds of primitive operators: computational operators and feature generation operators. For computational operators, the output is an image. For feature generation operators, however, the resultant output includes an image and a real number or vector. The real number or the vectors are the elements of the feature vector, which is used for classification. Table 1 shows different primitive operators and explains the meaning of each one (Tan et al., 2003; Bhanu et al., 2004).

The fitness value: During training, at every generation for each composite operator run by GP, we compute the feature vectors and input the feature vectors for all training images to train SVM classifier. We use C-Support Vector Classification (C-SVC) with RBF Kernel (Chang and Lin, 2001). Given a set of training vectors $x_i \in R^n$, $i = 1,\ldots,l$, belonging to two classes and $y \in R^l$. C-SVC solves the following problem:

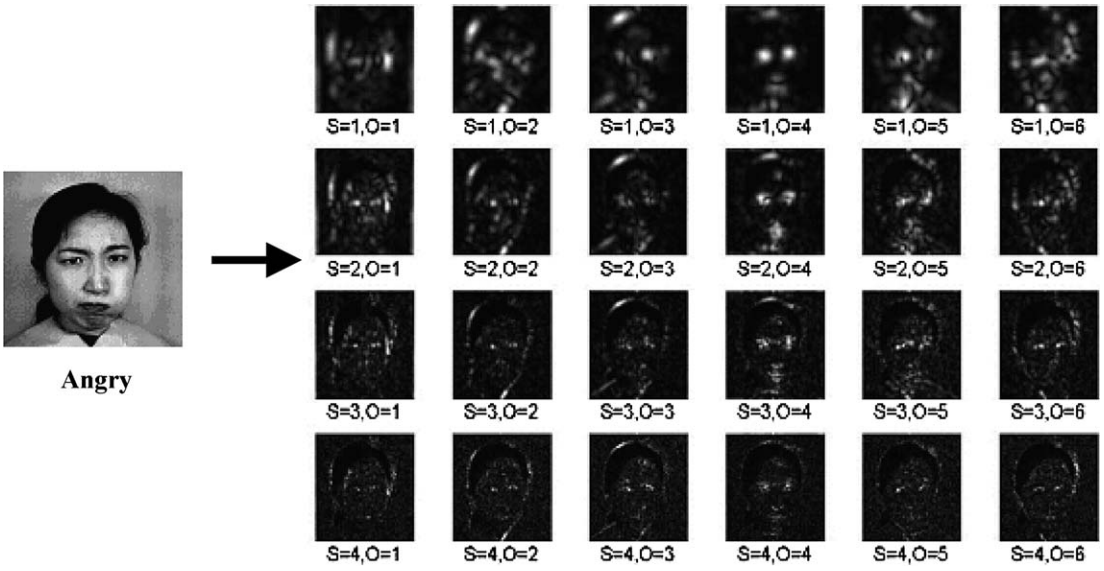

Fig. 4. An example of the primitive features (S means scale and O represents orientation), left image is the original facial expression image.

$$\min_{\omega,b,\xi} = \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{l}\xi_i$$
$$y_i(\omega^{\mathrm{T}}\phi(x_i) + b) \geqslant 1 - \xi_i, \quad \xi_i \geqslant 0, \ \ i = 1, \ldots, l$$

where ω, C and ξ_i are weight vector, constant and nonnegative bias variables, respectively. b is a scalar and $\phi(x_i)$ transforms an unknown vector to the separating space ($x_i \to \phi(x_i)$).

This primal problem can be transformed to its dual problem:

$$\min_{\alpha}\left(\frac{1}{2}\alpha^{\mathrm{T}}Q\alpha - e^{\mathrm{T}}\partial\right), \quad \alpha_i \in [0, C], \ \ i = 1, \ldots, l; \ y^{\mathrm{T}}\alpha = 0$$

where e is the vector of all ones, Q is an $l \times l$ positive semidefinite matrix, $Q_{ij} = y_i y_j K(x_i, x_j)$, and $K(x_i, x_j) \equiv \phi(x_i)^{\mathrm{T}}\phi(x_i)$ is the kernel. Here training vectors x_i are mapped into a higher (maybe infinite) dimensional space by the function ϕ. In this paper we use RBF kernel $K(x_i, x_j) = \mathrm{e}^{-\gamma\|x_i - x_j\|^2}$. The RBF kernel nonlinearly maps samples into a higher dimensional space, so it can handle the case when the relation between class labels and attributes is nonlinear (Chang and Lin, 2001). The decision function is

$$\mathrm{sign}\left(\sum_{i=1}^{l} y_i\alpha_i K(x_i, x) + b\right)$$

Given training samples, SVM can easily achieve high training accuracy (i.e., classifiers accurately predict training data whose class labels are indeed known). Since we use the fitness value to control the run of our GP, we do not want to overfit the training data and to terminate GP before it evolves a good composite operator. Cross-validation in our approach is used. We divide the database randomly into 10 roughly equal-sized parts, from which the data for nine parts are used for learning the features and training the classifier and the last part is used for testing. In the classification, the percentage of correct classification (PCC) is used as the fitness value of the composite operator.

$$\text{Fitness value} = \frac{n_{\mathrm{c}}}{n_{\mathrm{s}}} \times 100\%$$

where n_{c} is the number of correctly classified facial expression images by GP and n_{s} is the size of training set.

Parameters and termination conditions: The parameters to control the run of GP are important. In our approach, we select the maximum size of composite operator 200, population size 100, number of generation 50, crossover rate 0.6, length of maximum feature vector 35, the fitness threshold 0.98, and the mutation rate 0.05. GP stops whenever it finishes the pre-specified number of generations or whenever the best composite operator in the population has fitness value greater than the fitness threshold.

2.3. Multiple-agent approach

Multi-agent methodology can be used to boost the overall performance and compensate for the suboptimal character of representations elaborated by the evolutionary process (Krawiec and Bhanu, 2005). The basic prerequisite for the agents' fusion is their diversification. In our approach, agents correspond to classifiers; the diversification is naturally provided by the random nature of the genetic search. We run 10 genetic searches that start from different initial populations. With the same parameters, 10 independent GP synthesis processes provide a statistical significance to the results. Each run starts with the same GP parameters, but with different, randomly created, initial population of feature synthesis programs. In classification, we use a voting strategy: each run is considered to be voting where votes can be cast for each of the testing images. In the end we take the majority vote for a testing image. Fig. 5 shows the architecture of the compound recognition system.

Table 1
The primitive operators in our approach

Type	No.	Primitive operator	Meaning
Computation operators	1	ADD_OP	$A+B$, $A-B$, $A\times B$ and A/B. If the pixel in B has value 0, the corresponding pixel in A/B takes the maximum pixel value in A
	2	SUB_OP	
	3	MUL_OP	
	4	DIV_OP	
	5	MAX2_OP	$\mathrm{Max}(A,B)$
	6	MIN2_OP	$\mathrm{Min}(A,B)$
	7	ADD_CONST_ OP	$A+c$
	8	SUB_CONST_OP	$A-c$
	9	DIV_CONST_OP	A/c
	10	MUL_CONST_OP	$A\times c$
	11	SQRT_OP	$\mathrm{sign}(A)\times\sqrt{\lvert A\rvert}$
	12	LOG_OP	$\mathrm{sign}(A)\times\log(\lvert A\rvert)$
	13	MAX_OP	$\mathrm{Max}(A)$, $\mathrm{min}(A)$, $\mathrm{med}(A)$, $\mathrm{mean}(A)$ and $\mathrm{std}(A)$, replace the pixel value by the maximum, minimum, median, mean or standard deviation in a 3×3 block
	14	MIN_OP	
	15	MED_OP	
	16	MEAN_OP	
	17	STD_OP	
	18	BINARY_ZERO_OP	Threshold/binarize A by zero or mean of A
	19	BINARY_MEAN_OP	
	20	NEGATIVE_OP	$-A$
	21	LEFT_OP	Left (A), right (A), up (A) and down (A). Move A to the left, right, up or down by 1 pixel. The border is padded by zeros
	22	RIGHT_OP	
	23	UP_OP	
	24	DOWN_OP	
	25	HF_DERIVATIVE_OP	HF (A) and VF (A). Sobel filters along horizontal and vertical directions
	26	VF_DERIVATIVE_OP	
Feature generation operators	27	SPE_MAX_OP	$\mathrm{Max2}(A)$,
	28	SPE_MIN_OP	$\mathrm{Min2}(A)$
	29	SPE_MEAN_OP	$\mathrm{Mean2}(A)$
	30	SPE_ABS_MEAN_OP	$\mathrm{Mean2}(\lvert A\rvert)$
	31	SPE_STD_OP	$\mathrm{Std2}(A)$
	32	SPE_U3_OP	$\mu_3(A)$ and $\mu_4(A)$. Skewness and kurtosis of the histogram of A
	33	SPE_U4_OP	
	34	SPE_CENTER_MOMENT11_OP	$\mu_{11}(A)$. First order central moments of A
	35	SPE_MOMENT01_OP	Fist order moments of A
	36	SPE_MOMENT10_OP	Second order moments of A
	37	SPE_MOMENT11_OP	Third order moments of A
	38	SPE_ENTROPY_OP	$H(A)$. Entropy of A
	39	SPE_MEAN_VECTOR_OP	Mean_vector(A) and Std_vector(A). A vector contains the mean or standard deviation value of each row/column of A
	40	SPE_STD_VECTOR_OP	

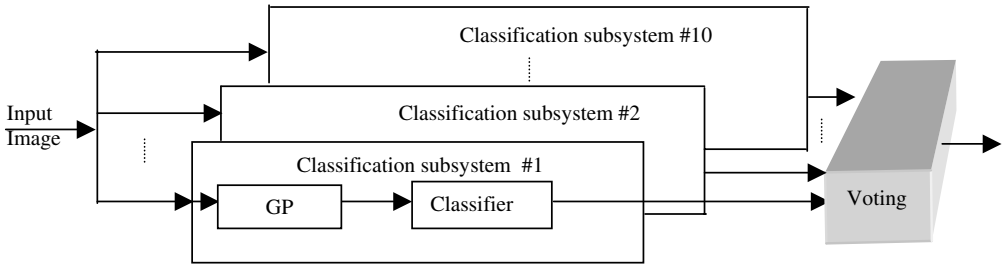

Fig. 5. Architecture of the compound classification system.

3. Experimental results

3.1. Database

The database we use for our experiments contains 213 images of 10 Japanese women (Lyons et al., 1998). Each person has two to four images for each of the seven expressions: neutral (30 images), happy (31 images), sad (31 images), surprise (30 images), anger (30 images), disgust (29 images), and fear (32 images). The size of each image is 256×256 pixels, which are downscaled to 32×32 for computational efficiency reasons. We divide the database

randomly into 10 roughly equal-sized parts, from which the data for nine parts are used for learning the features and training the classifier and the last part is used for testing which includes 21 images. A few examples are shown in Fig. 6. This database was also used in (Yacoob and Davis, 1994; Bartlett et al., 1996; Zhang, 1999; Guo and Dyer, 2003).

3.2. Results

Fig. 7 presents fitness charts of the best individuals from 10 runs. Each of these 10 evolved composite operators is computed on the test data. In classification, we use the voting strategy as described previously: each run is considered to be voting where votes are cast for all the testing images. At the end we take a majority vote for the testing image. In case that the two classes have identical votes, though it may not be a good strategy, we simply select the one with the largest index (we use indices from 0 to 6 as labels of the seven classes). After the majority votes, we get the classification rate at 80.95%, i.e., there are four failures in testing among 21 testing images. Table 2 shows the confusion matrix of the testing result. From the confusion matrix, we can find the error for class 5, which is fear expression, is the highest. For the database we use, Lyons et al. (1998) generated it and considers fear expression to be a

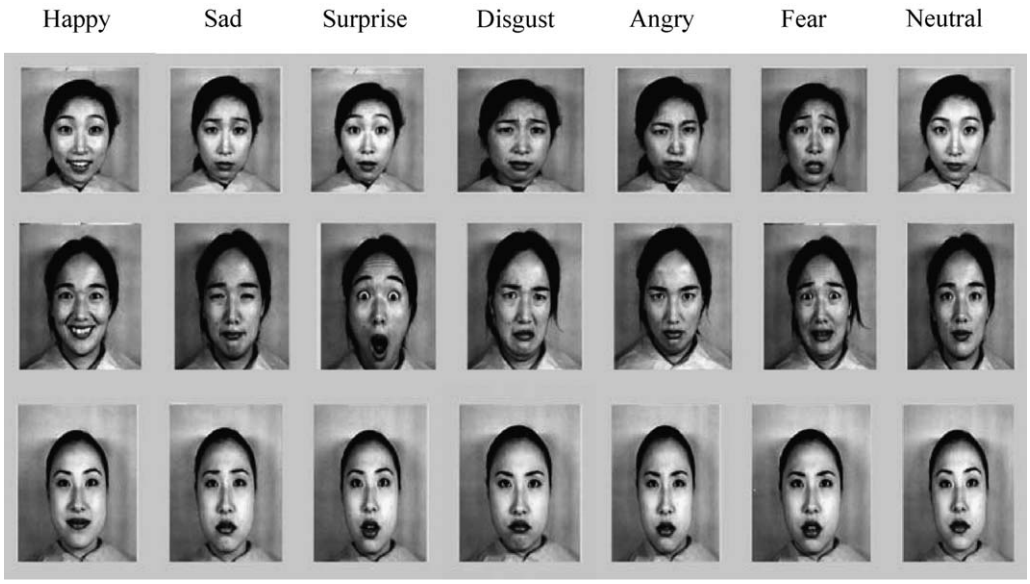

Fig. 6. A few example of the database.

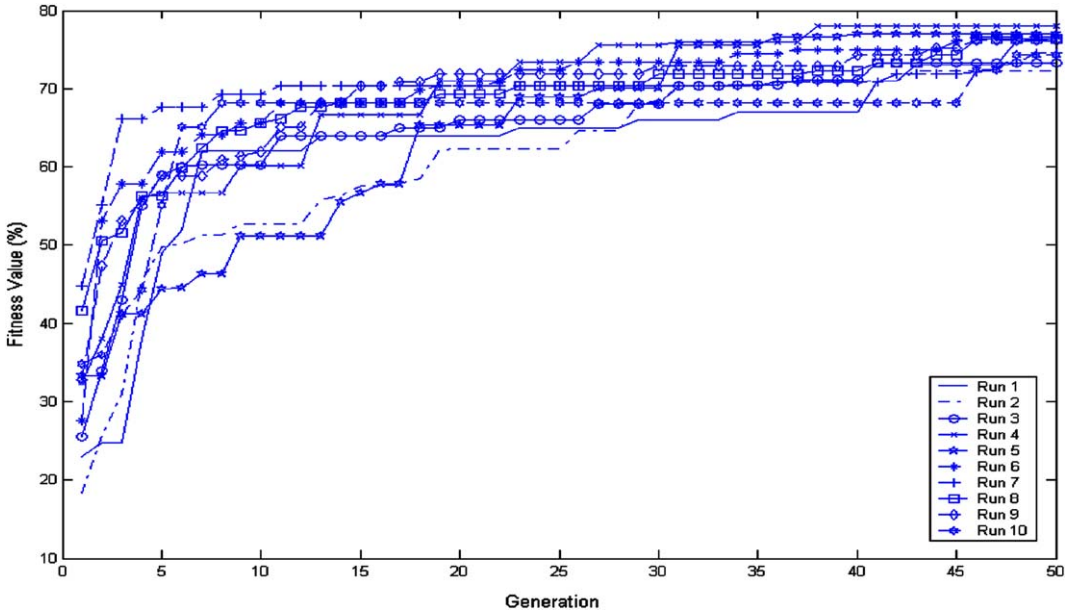

Fig. 7. Fitness charts of the best individuals with 10 runs.

Table 2
The confusion matrix for the testing

Input	Classified as						
	Happy	Sad	Surprise	Angry	Disgust	Fear	Neutral
Happy	**3**	0	0	0	0	0	0
Sad	1	**2**	0	0	0	0	0
Surprise	0	0	**3**	0	0	0	0
Angry	0	0	0	**2**	1	0	0
Disgust	0	0	0	0	**3**	0	0
Fear	0	0	0	0	1	**1**	**1**
Neutral	0	0	0	0	0	0	**3**

problematic expression for Japanese females because they are not good at posing fear expression. Fig. 8 shows the images of the four failures in testing. Ten best composite operators are learned in 10 runs. Fig. 9 shows one of the 10 evolved composite operators in Lisp notation. After computing the numbers of each computational primitive operator and feature generation primitive operator among the total numbers of two kinds of internal nodes (representing total computation primitive operators and total feature generation primitive operators), we get the frequency of occurrence of computational primitive operators and feature generation primitive operators, which are shown in Fig. 10(a) and (b). They show the total number of times each primitive operator is used in the 10 runs. We run our experiments on SunOS Ultra-60 workstation with 450 MHz CPU and 512 Megabytes memory. During training step, since we use GP and 10-fold cross-validation runs on training set to train SVM classifier, the experiments run slowly. Usually, it takes about 4 h to evolve one generation. However, once the training is finished, execution of composite operators is simple and it runs fast. On the average run-time for one testing image is 5 s.

3.3. Comparison with other methods

Guo and Dyer (2003) manually marks 34 fiducial points by hand and use various methods (see Table 2) on the same database that we use. Thus, for each image, the extracted feature vector is of dimension 612 ($34 \times 3 \times 6$), where they select three scales and six orientations for Gabor filtering. In our previous work (Bhanu et al., 2004), we use GP with a Bayesian Classifier to perform the facial expression rec-

Fig. 8. The four failure images in testing, from left to right: sad is classified as happy, angry is classified as disgust, fear is classified as disgust and fear is classified as neutral.

((**SPE_CENTER_MOMENT11_OP**)((MIN_OP)((LEFT_OP)((MAX_OP)((**SPE_ABS_MEAN_OP**)((DOWN_OP)((V F_DERIVATIVE_OP)((SUB_OP)((INPUT_OP)((STDV_OP)((LOG_OP)((MAX2_OP)((SPE_STD_OP)((MIN2_OP)((**SPE_ABS_MEAN_OP**)((SUB_OP)((VF_DERIVATIVE_OP)((MUL_CONST_OP)((MIN_OP)((MIN2_OP)((MIN_O P)((MEAN_OP)((**SPE_MAX_OP**)((INPUT_OP: 20))))) ((**SPE_MIN_OP**) ((**SPE_MOMENT10_OP**) ((SUB_OP) ((INPUT_OP: 22)((INPUT_OP: 18)))))))) ((**SPE_MEAN_OP**) ((**SPE_STD_OP**)((DIV_OP)((MUL_CONST_OP) ((MUL_OP)((MIN_OP)((ADD_OP)((BINARY_MEAN_OP) ((MIN_OP) ((SUB_CONST_OP) ((**SPE_MAX_OP**) ((BINARY_MEAN_OP) ((MAX2_OP) ((**SPE_MOMENT01_OP**) ((LOG_OP) ((**SPE_STD_OP**) ((INPUT_OP: 10))))) ((SUB_CONST_OP) ((**SPE_MAX_OP**) ((INPUT_OP: 23))))))))) ((MUL_CONST_OP) ((DOWN_OP) ((RIGHT_OP) ((HF_DERIVATIVE_OP) ((ADD_OP) ((UP_OP) ((SUB_OP) ((INPUT_OP: 3)) ((ADD_CONST_OP) ((MIN2_OP) ((DIV_OP) ((INPUT_OP: 22)) ((INPUT_OP: 0))) ((LEFT_OP)((INPUT_OP: 22))))))) ((MEAN_OP) ((HF_DERIVATIVE_OP) ((SUB_CONST_OP) ((**SPE_MAX_OP**) ((BINARY_MEAN_OP) ((MAX2_OP) ((**SPE_MOMENT01_OP**) ((ADD_CONST_OP) ((MIN2_OP) ((DIV_OP) ((INPUT_OP: 22) ((INPUT_OP: 0))) ((LEFT_OP) ((ADD_CONST_OP) ((MIN2_OP) ((DIV_OP) ((INPUT_OP: 22)) ((INPUT_OP: 0))) ((LEFT_OP) ((INPUT_OP: 22)))))))) ((SUB_CONST_OP) ((**SPE_MAX_OP**) ((UP_OP) ((SQRT_OP) ((MUL_CONST_OP) ((SUB_CONST_OP) ((**SPE_MOMENT01_OP**) ((DOWN_OP) ((MED_OP) ((**SPE_CENTER_MOMENT11_OP**) ((SQRT_OP) ((**SPE_STD_OP**) ((INPUT_OP: 0))))))))))))))))))))))))) ((INPUT_OP: 10))))) ((**SPE_MAX_OP**) ((LOG_OP) ((SUB_OP) ((SPE_STD_OP) ((INPUT_OP: 23))) ((ADD_CONST_OP) ((BINARY_MEAN_OP) ((MAX_OP) ((MUL_OP) ((LEFT_OP) ((MIN_OP) ((SUB_CONST_OP) ((**SPE_MAX_OP**) ((BINARY_MEAN_OP) ((MAX2_OP) ((**SPE_MOMENT01_OP**) ((LOG_OP) ((**SPE_STD_OP**) ((INPUT_OP: 10))))) ((SUB_CONST_OP)((**SPE_MAX_OP**) ((INPUT_OP: 23))))))))) ((SQRT_OP) ((**SPE_MIN_OP**) ((**SPE_MIN_OP**) ((MUL_OP) ((INPUT_OP: 3)) ((**SPE_MAX_OP**) ((INPUT_OP: 20))))))))))))))))))) ((INPUT_OP: 9)))) ((HF_DERIVATIVE_OP) ((SQRT_OP) ((**SPE_STD_OP**) ((**SPE_MAX_OP**) ((BINARY_MEAN_OP) ((MAX2_OP) ((**SPE_MOMENT01_OP**) ((LOG_OP) ((INPUT_OP: 6)))) ((SUB_CONST_OP) ((**SPE_MAX_OP**) ((UP_OP) ((SQRT_OP) ((MUL_CONST_OP) ((SUB_CONST_OP) ((**SPE_MOMENT01_OP**) ((DOWN_OP) ((MED_OP) ((**SPE_CENTER_MOMENT11_OP**) ((SQRT_OP) ((**SPE_STD_OP**) ((DOWN_OP) ((INPUT_OP: 6)))))))))))))))))))))))))))))))

Fig. 9. One of the learned composite operators, size is 151 (feature generation primitive operators are shown in bold).

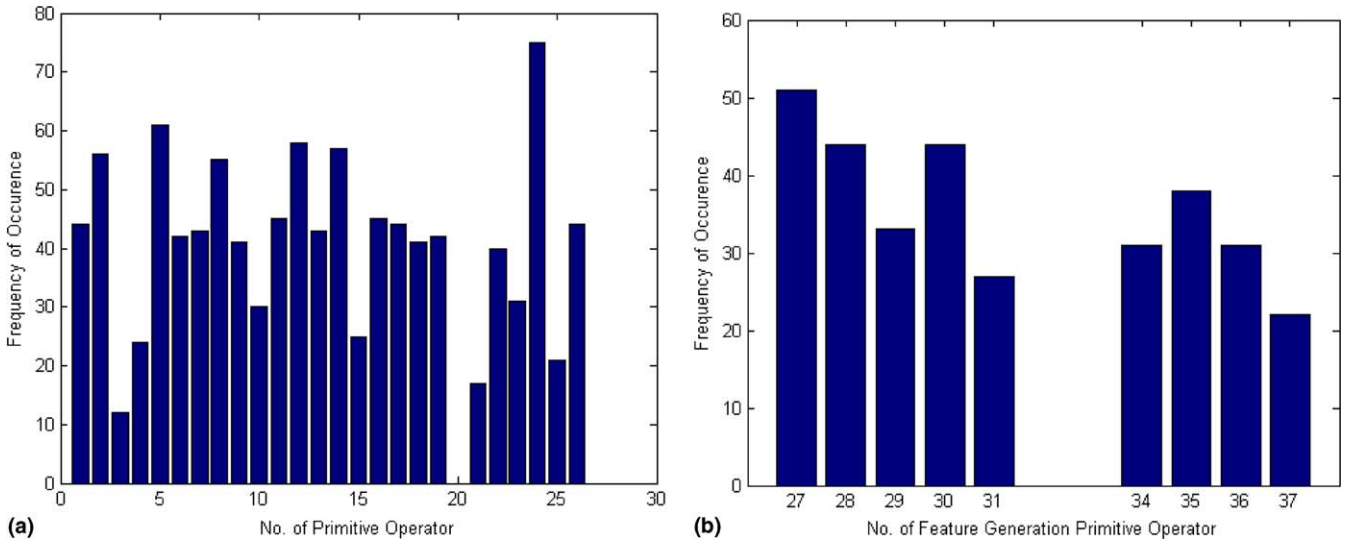

Fig. 10. Frequency of occurrence of primitive operators. (a) Frequency of occurrence of computational primitive operators. (b) Frequency of occurrence of feature generation primitive operators.

Table 3
Comparison of the recognition accuracy

	Bayes All (Guo and Dyer, 2003)	Bayes FS (Guo and Dyer, 2003)	AdaBoost (Guo and Dyer, 2003)	Bayes GP (Bhanu et al., 2004)	NL-SVM (Guo and Dyer, 2003)	GP (this paper)
Feature selection	Hand	Hand	Hand	Automatic	Hand	Automatic
Accuracy	63.3%	71.0%	71.9%	71%	91.9%	80.95%
# Features	612	60	80	25	612	35

ognition. We compare all the results in Table 3. In Table 3, 'Bayes All' means the Bayes classifier without feature selection, which means 612 dimensional feature vector (Guo and Dyer, 2003). 'Bayes FS' means Bayes classifier with pairwise-greedy feature selection (Guo and Dyer, 2003). 'NL-SVM' means SVM classifier with nonlinear kernel (Guo and Dyer, 2003). In our approach, we did not do any pre-processing of the raw images. The input image is the raw facial expression image. However, the other methods in Table 3 selected the fiducial points on a face image manually and generated the Gabor coefficients as a feature vector.

4. Conclusion

In this paper, we propose a learning algorithm for facial expression recognition based on GP. The proposed approach learns feature vector for facial expression recognition without explicit estimation of object pose, without any hand-tuned pre-process specific to a database. Thus, our approach is automatic and database-independent. Compared to the previous work, our experimental results show that GP can find good composite operators. Our GP-based algorithm is effective in extracting feature vectors for classification. In our approach, we do not need to perform any pre-processing of the raw image and we do not need to find any reference points on the face. The effectiveness of synthesized composite feature is dependent on the effectiveness of primitive feature and the primitive operators. It will be difficult for GP to evolve effective features if primitive features do not capture the most significant characteristics of the facial expression images. Thus, it is important to design effective primitive features. Without any pre-processing on the facial expression images, our approach is able to synthesize composite features in an automatic manner and provides good classification results.

References

Bartlett, M., Viola, P., Sejnowski, T., Larsen, L., Hager, J., Ekman, P., 1996. Classifying facial action. In: Touretzky, D., Mozer, M., Hasselmo, M. (Eds.), Advances in Neural Information Processing Systems, vol. 8. MIT Press, Cambridge, MA.

Bhanu, B., Yu, J., Tan, X., Lin, Y., 2004. Feature synthesis using genetic programming for face expression recognition. Genetic Evol. Comput. Conf. 2, 896–907.

Bhanu, B., Lin, Y., Krawiec, K., 2005. Evolutionary Synthesis of Pattern Recognition Systems. Springer.

Chang, C., Lin, C., 2001. LIBSVM: A library for support vector machines. Available from: <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.

Daugman, J., 1997. Face and gesture recognition: An overview. IEEE Trans. Pattern Anal. Machine Intell. 19 (7), 675–676.

Guo, G.D., Dyer, C.R., 2003. Simultaneous feature selection and classifier training via linear programming: A case study for face expression recognition. IEEE Conf. Computer Vision Pattern Recogn. 1, 346–352.

Hoey, J., Little, J.J., 2000. Representation and recognition of complex human motion. IEEE Conf. Computer Vision Pattern Recogn. 1, 752–759.

Jain, A.K., Farokhnia, F., 1991. Unsupervised texture segmentation using Gabor filters. Pattern Recognition 16 (12), 1167–1186.

Koza, J., 1994. Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press.

Krawiec, K., Bhanu, B., 2005. Evolutionary feature synthesis for visual learning. IEEE Trans. Systems Man Cybernet. 35 (3), 409–425.

Lyons, M.J., Akamatsu, S., Kamachi, M., Gyoba, J., 1998. Coding facial expressions with Gabor wavelets. In: Proc. Third IEEE Int. Conf. Automatic Face and Gesture Recognition, 1998, pp. 200–205.

Lyons, M.J., Budynek, J., Akamatsu, S., 1999. Automatic classification of single facial images. IEEE Trans. Pattern Anal. Machine Intell. 21 (12), 1357–1362.

Manjunath, B.S., Ma, W.Y., 1996. Texture features for browsing and retrieval of image data. IEEE Trans. Pattern Anal. Machine Intell. 18 (8), 837–842.

Pantic, M., Rothkrantz, L.J.M., 2000. Automatic analysis of facial expressions: the state of the art. IEEE Trans. Pattern Anal. Machine Intell. 22, 1424–1445.

Tan, X., Bhanu, B., Lin, Y., 2003. Learning features for fingerprint classification. In: Proc. Conf. on Audio- and Video-based Biometric Person Authentication, 2003, pp. 319–326.

Viola, P., Jones, M., 2001. Rapid object detection using a boosted cascade of simple features. IEEE Conf. Computer Vision Pattern Recogn. 1, 511–518.

Wiskott, L., Fellous, J.M., Kruger, N., Von der Malsburg, C., 1997. Face recognition by bunch graph matching. IEEE Trans. Pattern Anal. Machine Intell. 19 (7), 775–779.

Wu, X., Bhanu, B., 1997. Gabor wavelet representation for 3-D object recognition. IEEE Trans. Image Process. 6 (1), 47–64.

Yacoob, Y., Davis, L., 1994. Recognizing facial expressions by spatio-temporal analysis. In: Proc. Internat. Conf. on Pattern Recognition, vol. 1, 1994, pp. 747–749.

Zhang, Z., 1999. Feature-based facial expression recognition: Sensitivity analysis and experiments with a multi-layer perceptron. J. Pattern Recogn. Artificial Intell. 13 (6), 893–911.

Zhang, Z., Lyons, M., Schuster, M., Akamatsu, S., 1998. Comparison between geometry-based and Gabor-wavelets-based facial expression recognition using multi-layer perceptron. In: Proc. Int. Conf. Automatic Face and Gesture Recognition, 1998, pp. 454–459.