Learning From Examples in the Small Sample Case: Face Expression Recognition

Guodong Guo and Charles R. Dyer, *Fellow, IEEE*

***Abstract*—Example-based learning for computer vision can be difficult when a large number of examples to represent each pattern or object class is not available. In such situations, learning from a small number of samples is of practical value. To study this issue, the task of face expression recognition with a small number of training images of each expression is considered. A new technique based on linear programming for both feature selection and classifier training is introduced. A pairwise framework for feature selection, instead of using all classes simultaneously, is presented. Experimental results compare the method with three others: a simplified Bayes classifier, support vector machine, and AdaBoost. Finally, each algorithm is analyzed and a new categorization of these algorithms is given, especially for learning from examples in the small sample case.**

***Index Terms*—AdaBoost, Bayes decision, face expression recognition, feature selection, large margin classifiers, learning by example, linear programming, Gabor wavelets, small sample case, statistical learning, support vector machine.**

I. Introduction

EXAMPLE-BASED learning in computer vision and pattern recognition is an important problem, however, it is sometimes not easy to collect a large number of examples to represent each pattern or object class. Hence, learning in the small sample case is of practical interest. One reason for this is the difficulty in collecting image data for each object. For example, in face recognition or face expression recognition (FER), it is not easy to collect a database with a large number of images of each individual or expression. Recently, some approaches use only one training example for face recognition [22]. In image retrieval, the user can give some positive and negative examples based on relevance feedback [10], but it is impractical to expect the user to provide a large number of images. Furthermore, it is often difficult to collect training data to cover all possible cases. For instance, in face recognition, it is well known that there are many factors, including illumination, expression, pose, facial hair, eye glasses, and age that influence the image of a face. Obviously, it is not practical to collect images covering every possible situation. Consequently, learning from a small, incomplete set of samples is an essential issue in computer vision. In this paper we use FER as an example task to study this problem.

Manuscript received July 31, 2003; revised April 1, 2004. This work was supported by the National Science Foundation (NSF) under Grant IIS-9 988 426 and by the University of Wisconsin. This paper was recommended by Associate Editor B. Bhanu.

The authors are with the Computer Sciences Department, University of Wisconsin-Madison, Madison, WI 53706 USA.

Digital Object Identifier 10.1109/TSMCB.2005.846658

A. Face Expression Recognition

FER by computer is useful for many applications such as human behavior understanding, perceptual user interfaces, and interactive computer games. In an automatic FER system, the first step is face detection or localization in a cluttered scene. Next, relevant features from the face must be extracted, and finally the expression can be classified based on the extracted features. Unlike face recognition, FER focuses on how to discern the same expressions from different individuals. Because different people may show the same expression in different ways, the FER problem is more challenging.

There are two versions of the FER problem depending on whether an image sequence is the input and the dynamic characteristics of expressions are analyzed, or a single image is the input and expressions are distinguished based on static differences. See [25] for a review of different approaches for FER. Here, we are interested in FER from single images. Previous work by Padgett and Cottrell [24] used seven pixel blocks from feature regions to represent expressions. Cottrell and Metcalfe [4] used principal component analysis and feed-forward neural networks. Rahardja *et al.* [27] used a pyramid structure with neural networks. Lanitis *et al.* [16] used parameterized deformable templates to represent face expressions. Lyons *et al.* [18], [19] and Zhang *et al.* [38], [37] demonstrated the advantages of using Gabor wavelet coefficients to code face expressions.

In this paper we use Gabor filters for facial feature extraction. Our major focus is on the evaluation of some new methods for FER. Recently, large margin classifiers such as support vector machines (SVMs) [32] and AdaBoost [7], [29] were studied in the machine learning community, and have been used for solving some vision problems. Here, we are interested to see if they are useful for FER learning in the small sample case. To our knowledge, this is the first time large margin classifiers have been evaluated for FER.

B. Feature Selection in Pattern Recognition

The goal of feature selection is to preprocess the image data to obtain a small set of the most important properties while retaining the salient characteristics of the data. The benefits of feature selection are not only to reduce recognition time by reducing the amount of data that needs to be analyzed, but also, in many cases, to produce better classification accuracy due to finite sample size effects [14].

Most feature selection methods involve evaluating different feature subsets using some criterion such as probability of error [14]. One difficulty with this approach when applied to real

problems with large feature dimensionality, is the high computational complexity involved in searching the exponential space of feature subsets. Jain and Zongker [14] evaluated different search algorithms for feature subset selection and found that the Sequential Forward Floating Selection (SFFS) algorithm proposed by Pudil *et al.* [26] performed best. However, SFFS is very time consuming when the number of features is large. For example, Vailaya [31] used the SFFS method to select 67 features from 600 for a two-class problem and reported that SFFS required 12 days of computation time.

Another issue associated with feature selection methods is the *curse of dimensionality*, i.e., the problem of feature selection when the number of features is large but the number of samples is small [14]. This situation is common in many computer vision tasks such as object recognition because there are often less than tens of training samples (images) for each object, but there are hundreds of candidate features extracted from each image.

Yet another associated problem is determining how many features to select for a given data set. Traditional feature selection methods do not address this problem and require the user to choose the number of features. Consequently, this parameter is usually set without a sound basis.

The AdaBoost[1] method formulated by Tieu and Viola [30] in image retrieval can heuristically select a subset of features while learning the classifier. SVM cannot select features in its standard dual formulation [32]. A simplified Bayes classifier can do feature selection in a heuristic way by assuming the features are independent and all classes have the same covariance matrix in order to simplify the density estimation problem for each class in the small sample case. However, the heuristic strategy in AdaBoost and the simplified Bayes classifier cannot truly solve the feature selection problem. We will introduce a novel algorithm based on linear programming (LP) that can address the feature selection problem simultaneously in classifier training without using heuristics such as those used in the simplified Bayes classifier or AdaBoost.

The organization of the paper is as follows. In Section II, the Gabor filter bank design and feature extraction methods are described. In Section III, three representative classifiers are presented. Section IV introduces a novel algorithm called FSLP and analyzes why it can select features with a sound basis. In Section V, we argue that a pairwise feature selection framework is superior to selection using all classes. After presenting experimental results in Section VI, we give a general analysis of learning techniques in Section VII.

II. Facial Feature Extraction

For FER it is common to detect some fiducial points and then compute features at these points instead of using the whole image. Given a set of fiducial points detected or marked on a face image, two approaches to facial feature extraction are to use either: 1) the geometric positions of the fiducial points or 2) the Gabor filter coefficients [5] at the fiducial points. It has been shown [19], [38] that filter coefficients characterize face expressions better than geometric positions. Therefore, in our study, we use Gabor filtering for facial feature extraction.

[1]The AdaBoost method [7] is a meta-algorithm for classification or regression that adaptively boosts a series of weak-learners to make the final decision. A variant of AdaBoost [30] lets each weak-learner work on a single feature and thus can also do feature selection. We refer to this variant of AdaBoost in this paper.

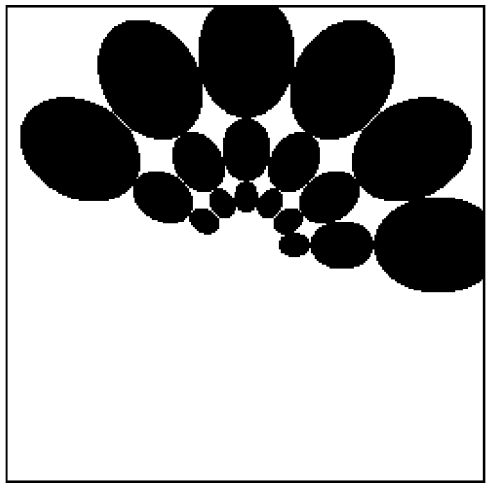


Fig. 1. Filter set in the spatial-frequency domain. There are a total of 18 Gabor filters shown at half-peak magnitude.

A. The Gabor Filter Bank

A two-dimensional Gabor function $g(x,y)$ and its Fourier transform $G(u,v)$ can be written as

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

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

where W is the frequency of a sinusoidal plane wave along the x-axis, and σ_x and σ_y are the space constants of the Gaussian envelope along the x and y axes, respectively. $\sigma_u = 1/2\pi\sigma_x$ and $\sigma_v = 1/2\pi\sigma_y$. Filtering a signal with this basis provides a localized frequency characterization. Filters with arbitrary orientations can be obtained by a rigid rotation of the x-y coordinate system

$$g'(x,y) = g(x',y') \quad (3)$$

where

$$x' = x\cos\theta + y\sin\theta, \quad y' = -x\sin\theta + y\cos\theta \quad (4)$$

and θ is the rotation angle.

In earlier applications of Gabor filtering [5] for face recognition [15], [36] and face expression classification [18], [19] [38], [37], investigators have only varied the scale and orientation of the filters, but kept the Gaussian envelope parameter σ fixed to π or 2π. This methodology is questionable because the area of the energy distribution of the filters varies with scale, so the Gaussian envelope should vary with the filter size. Consequently, we designed the Gabor filter bank based on the filters used perviously for texture segmentation and image retrieval [13], [21].

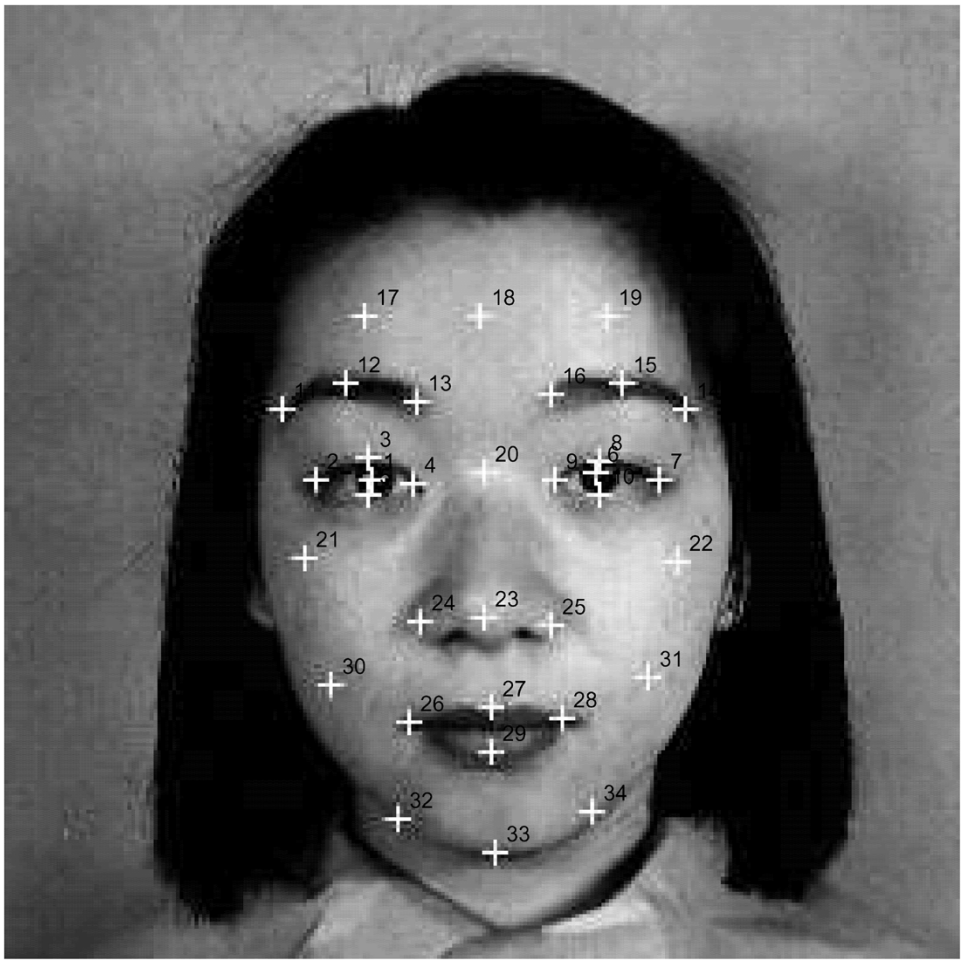


Fig. 2. Thirty-four fiducial points on a face image.

The Gabor filter bank is designed to cover the entire frequency spectrum [13], [21]. In other words, the Gabor filter set is constructed such that the half-peak magnitude of the filters in the frequency spectrum touch each other. This results in the following formulas to compute the filter parameters σ_u and σ_v:

$$a = \left(\frac{U_h}{U_l}\right)^{\frac{1}{S-1}}, \quad W = a^m U_l \tag{5}$$

$$\sigma_u = \frac{(a-1)W}{(a+1)\sqrt{2\ln 2}} \tag{6}$$

$$\sigma_v = \tan\left(\frac{\pi}{2K}\right)\left[W - \frac{(2\ln 2)\sigma_u^2}{W}\right] \times \left[2\ln 2 - \frac{(2\ln 2)^2\sigma_u^2}{W^2}\right]^{-\frac{1}{2}} \tag{7}$$

where U_l and U_h denote the lower and upper center frequencies of interest. $m \in \{0, 1, \ldots, S-1\}$ and $n \in \{0, 1, \ldots, K-1\}$ are the indices of scale and orientation, respectively. K is the number of orientations and S is the number of scales.

In our experiments, we used $U_h = \sqrt{2}/4, U_l = \sqrt{2}/16$, three scales ($S = 3$), and six orientations ($K = 6$). The half-peak support of the Gabor filter bank is shown in Fig. 1. The differences in the strength of the responses of different image regions is the key to the multichannel approach to face image analysis.

B. Feature Extraction

After Gabor filtering, the amplitude values at selected fiducial points on the face images are used as the features. Automatically extracting these points [15], [36] is still an open problem [38]. In order to focus this study on the classifier performance, we manually marked the fiducial points in each image. Typical positions of 34 fiducial points are shown in Fig. 2. Thus, for each face image, the extracted feature vector is of dimension 612 ($=34 \times 3 \times 6$), where three scales and six directions for Gabor filtering are used.

C. The FER System

Our approach to FER is summarized in Fig. 3. Each input face image is convolved with 18 Gabor filters and results in 18 filtered images. The amplitude of each filtered image at selected fiducial points are used as feature values. The feature vector is

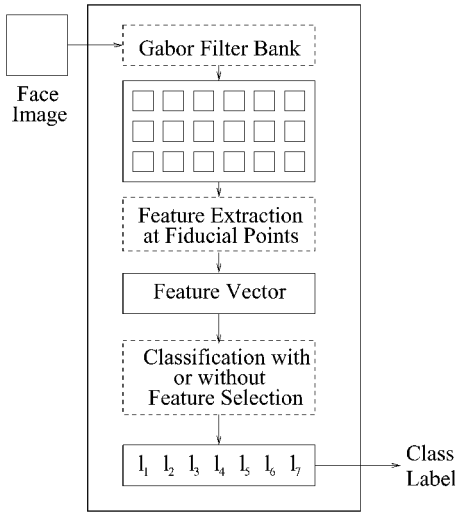


Fig. 3. Framework of our FER system. Dashed blocks represent operations while solid blocks contain the filtered images, feature vector, and class labels.

used for classification with different classifiers. In the learning stage, some classifiers can also perform feature selection.

III. Classifiers

In this section, three standard classifiers for FER are described. The first is the Bayes classifier [8], which is optimal when the probability distribution is known for each class. To use the Bayes classifier for the small sample case, one has to make some assumptions to simplify the density estimation problem given the training data [17]. We call this the simplified Bayes classifier in this paper. The second classifier is the SVM, and the third is the AdaBoost method (recall that it is a variant of the AdaBoost [7]). The latter two are called large margin classifiers in the machine learning community [29]. In the remainder of this section, relevant details on these three classifiers are reviewed. A new algorithm based on LP is introduced in Section IV.

A. A Simplified Bayes Classifier

The Bayes classifier yields minimum error rates when the underlying probability density function (pdf) is known [8]. The *a posteriori* probability of pattern $\mathbf{x}$ belonging to class ω_c is given by Bayes' rule

$$P(\omega_c \,|\, \mathbf{x}) = \frac{P(\omega_c)p(\mathbf{x} \,|\, \omega_c)}{p(\mathbf{x})} \tag{8}$$

where $P(\omega_c)$ is the *a priori* probability, $p(\mathbf{x} \,|\, \omega_c)$ the conditional probability density function of ω_c, and $p(\mathbf{x})$ is the mixture density. The maximum *a posteriori* (MAP) decision is

$$\omega_c^* = \arg\max_c P(\omega_c \,|\, \mathbf{x}), \quad c = 1, 2, \ldots, C, \tag{9}$$

The Bayes classifier can be used for both two-class and multiclass classifications.

In FER, there are often not enough training images of each expression to reliably estimate the conditional density function for each class. A compromise is to assume that the within-class densities can be modeled as normal distributions, and all the within-class covariance matrices are identical and diagonal. Liu and Wechsler [17] used this simplification for face recognition. Here we evaluate this approach for the problem of FER. The parameters of the normal distributions are estimated as

$$\mu_c = \frac{1}{N_c}\sum_{j=1}^{N_c} \mathbf{x}_j^{(c)}, \quad c = 1, 2, \ldots, C \tag{10}$$

where $\mathbf{x}_j^{(c)}, j = 1, 2, \ldots, N_c$ represents the samples from class ω_c and

$$\Sigma_I = \Sigma_c = \mathrm{diag}\left\{\sigma_1^2, \sigma_2^2, \ldots, \sigma_D^2\right\} \tag{11}$$

where D is the feature dimension. Each component σ_i^2 can be estimated by the sample variance in the one-dimensional feature subspace

$$\sigma_i = \frac{1}{C}\sum_{c=1}^{C}\left\{\frac{1}{N_c - 1}\sum_{j=1}^{N_c}\left(x_{ji}^{(c)} - \mu_{ci}\right)^2\right\} \tag{12}$$

where $x_{ji}^{(c)}$ is the ith element of the sample $\mathbf{x}_j^{(c)}$, μ_{ci} the ith element of μ_c, and C the number of classes.

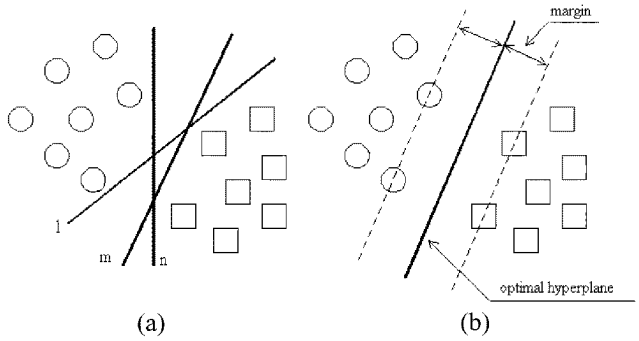


Fig. 4. Classification between two classes using hyperplanes. (a) Arbitrary hyperplanes l, m, and n. (b) The optimal separating hyperplane with the largest margin identified by the dashed lines, which go through the support vectors.

B. Support Vector Machine

Given a set of training vectors belonging to two separate classes, $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)$, where $\mathbf{x}_i \in R^n$ and $y_i \in \{-1, +1\}$, one wants to find a hyperplane $\mathbf{w}\mathbf{x} + b = 0$ to separate the data. Fig. 4(a) shows an example and several possible hyperplanes, but there is only one [shown in Fig. 4(b)] that maximizes the margin (i.e., the distance between the hyperplane and the nearest data point in each class). This linear classifier is called the optimal separating hyperplane (OSH).

The solution to the optimization problem of SVMs is given by the saddle point of the Lagrange functional

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l}\alpha_i\{y_i[(\mathbf{w}\cdot\mathbf{x}_i) + b] - 1\} \tag{13}$$

where α_i are the Lagrange multipliers. Classical Lagrangian duality enables the *primal* problem (13) to be transformed to its *dual* problem

$$\bar{\alpha} = \arg\max_{\alpha} \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (\mathbf{x_i} \cdot \mathbf{x_j}) \tag{14}$$

subject to

$$\alpha_i \geq 0 \; (i = 1, \ldots, l), \quad \sum_{i=1}^{l} \alpha_i y_i = 0 \tag{15}$$

which is easier to solve. The solution is given by

$$\bar{\mathbf{w}} = \sum_{i=1}^{l} \bar{\alpha}_i y_i \mathbf{x}_i, \quad \bar{b} = -\frac{1}{2} \bar{\mathbf{w}} \cdot [\mathbf{x}_r + \mathbf{x}_s] \tag{16}$$

where $\mathbf{x}_r$ and $\mathbf{x}_s$ are any two support vectors with $\bar{\alpha}_r, \bar{\alpha}_s > 0, y_r = 1$ and $y_s = -1$.

To solve a nonseparable problem, Cortes and Vapnik [3] introduced slack variables $\xi_i \geq 0$ and a penalty function, $F(\xi) = \sum_{i=1}^{l} \xi_i$, where the ξ_i measure the misclassification error. The solution is identical to the separable case except for a modification of the Lagrange multipliers as $0 \leq \alpha_i \leq M, i = 1, \ldots, l$. The choice of M is not critical in practice, and we used $M = 100$ in all our experiments. See [32] for more details on the nonseparable case.

SVMs can realize nonlinear discrimination by kernel mapping [32]. When the samples in the input space cannot be separated by any linear hyperplane, they may be linearly separated in a nonlinearly mapped feature space. Note that here the feature space of the SVMs is different from the image feature space.

Several kernel functions have been used previously for nonlinear mapping [32], with the Gaussian radial basis function (GRBF) the most commonly used. In our experiments we used a GRBF kernel of the form $K(x, y) = \exp(-((\mathbf{x}-\mathbf{y})^2)/(\gamma^2))$, where parameter γ is the width of the Gaussian function.

For a given kernel function, the SVM classifier is now given by

$$f(\mathbf{x}) = \operatorname{sign}\left(\sum_{i=1}^{l} \bar{\alpha}_i y_i K(\mathbf{x}_i, \mathbf{x}) + \bar{b} \right). \tag{17}$$

C. AdaBoost

Boosting is a method for combining a collection of weak classification functions (weak learners) to form a stronger classifier. AdaBoost is an adaptive algorithm that boosts a sequence of classifiers, in that the weights are updated dynamically according to the errors in earlier learning [7]. AdaBoost belongs to the class of large margin classifiers. The original AdaBoost method [7] works on all given features. Tieu and Viola adapted the AdaBoost algorithm for image retrieval [30] and for face detection [33], using a single feature at a time. Thus each weak learner uses a threshold to separate two classes, which is computed by the midpoint between the two classes in one dimension. After T rounds of boosting, T features are selected together with the T weak classifiers. Tieu and Viola's AdaBoost algorithm [30] is as follows.

AdaBoost Algorithm

Input: 1) n training examples, $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$, with $y_i = 1$ or 0

2) the number of iterations, T

Initialize weights $w_{1,i} = (1/2l)$ or $(1/2m)$ for $y_i = 1$ or 0, respectively, with $l + m = n$

Do for $t = 1, \ldots, T$:

1. Train one hypothesis h_j for each feature j with w_t, and error $\epsilon_j = Pr_i^{w_t}[h_j(x_i) \neq y_i]$

2. Choose $h_t(\cdot) = h_k(\cdot)$ such that $\forall_j \neq k, \epsilon_k < \epsilon_j$. Let $\epsilon_t = \epsilon_k$

3. Update: $w_{t+1,i} = w_{t,i}\beta_t^{e_i}$, where $e_i = 1$ or 0 for example x_i classified correctly or incorrectly, respectively, with $\beta_t = (\epsilon_t)/(1 - \epsilon_t)$ and $\alpha_t = \log(1/\beta_t)$

4. Normalize the weights so they are a distribution, $w_{t+1,i} \leftarrow (w_{t+1,i})/(\sum_{j=1}^{n} w_{t+1,j})$

Output the final hypothesis

$$h_f(x) = \begin{cases} 1 & \text{if } \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

IV. Discrimination and Feature Selection With Linear Programming

In contrast to the methods described in the last section, which mainly deal with the classification problem, here we introduce a new algorithm based on LP that can address both feature selection and classifier training simultaneously. Furthermore, we analyze why this LP technique circumvents the *curse of dimensionality* problem for feature selection in the small sample case, which is a novel and important result. Note that standard SVM cannot do feature selection, and the simplified Bayes and AdaBoost methods choose features heuristically but cannot determine how many (see Section V for details).

A. LP Formulation

In the early 1960s, the LP technique [20] was used to address the pattern separation problem. Later, a robust LP technique was proposed to deal with linear inseparability [1]. Recently, the LP framework has been extended to cope with the feature selection problem [2]. We briefly describe this new LP formulation below.

Given two sets of points, $\mathcal{A}$ and $\mathcal{B}$, in R^n, we seek a linear function such that $f(x) > 0$ if $x \in \mathcal{A}$ and $f(x) \leq 0$ if $x \in \mathcal{B}$. This function is given by $f(x) = w'x - \gamma$, and determines a plane $w'x = \gamma$ with normal $w \in R^n$ that separates points $\mathcal{A}$ from $\mathcal{B}$. Let the set of m points in $\mathcal{A}$ be represented by a matrix $A \in R^{m \times n}$, and the set of k points in $\mathcal{B}$ be represented by a matrix $B \in R^{k \times n}$. After normalization, we want to satisfy

$$Aw \geq e\gamma + e, \quad Bw \leq e\gamma - e \tag{19}$$

where e is a vector of all 1s with appropriate dimension. Practically, because of overlap between the two classes, one has to minimize some norm of the average error in (19) [1]

$$\min_{(\omega,\gamma)} f(\omega,\gamma) = \min_{\omega,\gamma} \frac{1}{m}\|(-Aw + e\gamma + e)_+\|_1 + \frac{1}{k}\|(Bw - e\gamma + e)_+\|_1 \tag{20}$$

where x_+ denotes the vector with components $\max\{0, x_i\}$. There are two main reasons for choosing the 1-norm in (20): 1) it is easy to formulate as a linear program (see (21) below) with theoretical properties that make it computationally efficient [1] and 2) the 1-norm is less sensitive to outliers such as those occurring when the underlying data distributions have pronounced tails [2].

Equation (20) can be modeled as a robust linear programming (RLP) problem [1], as follows:

$$\begin{aligned} \min_{w,\gamma,y,z} \quad & \frac{e'y}{m} + \frac{e'z}{k} \\ \text{subject to} \quad & -Aw + e\gamma + e \le y, \\ & Bw - e\gamma + e \le z, \\ & y \ge 0, \quad z \ge 0 \end{aligned} \tag{21}$$

which minimizes the average sum of misclassification errors of the points to two bounding planes, $x'w = \gamma+1$ and $x'w = \gamma-1$, where "$'$" represents transpose.

Problem (21) solves the classification problem without considering the feature selection problem. In [2] a feature selection strategy was integrated into the objective function in order to simultaneously select a subset of the features. Feature selection is defined by suppressing as many components of the normal vector w to the separating plane P as needed to obtain an acceptable discrimination between the sets $\mathcal{A}$ and $\mathcal{B}$. To accomplish this, an extra term is added to the objective function of (21), reformulating it as

$$\begin{aligned} \min_{w,\gamma,y,z} \quad & (1-\lambda)\left(\frac{e'y}{m} + \frac{e'z}{k}\right) + \lambda e'|w|_* \\ \text{subject to} \quad & -Aw + e\gamma + e \le y, \\ & Bw - e\gamma + e \le z, \\ & y \ge 0, \quad z \ge 0 \end{aligned} \tag{22}$$

where $|w|_* \in R^n$ has components equal to 1 if the corresponding components of w are nonzero, and has components equal to 0 if the corresponding components of w are 0. So, $e'|w|_*$ is actually a count of the nonzero elements in the vector w. This is the key to integrating feature selection with classifier training. Problem (22) balances the error in discrimination between two sets $\mathcal{A}$ and $\mathcal{B}$, $(e'y)/(m)+(e'z)/(k)$, *and* the number of nonzero elements of w, $e'|w|_*$. Moreover, if an element of w is 0, the corresponding feature is removed. Thus only the features corresponding to nonzero components in the normal w are selected after LP optimization.

Bradley and Mangasarian [2] developed a method called *feature selection via concave minimization* (FSV) to deal with the last term in the objective function of (22). They first introduced a variable v to eliminate the absolute value in the last term by replacing $e'|w|_*$ with $e'v_*$ and adding a constraint $-v \le w \le v$, which models the vector $|w|$. Because the step function $e'v_*$ is discontinuous, they used a concave exponential to approximate it, $v_* \approx t(v,\alpha) = e - \varepsilon^{-\alpha v}$, in order to get a smooth solution. This required introduction of an additional parameter, α. Alternatively, instead of computing the concave exponential approximation, we will use a simple term $e's$ with only one parameter, μ, and s is the component-wise absolute value of w. This produces our formulation, which we call *feature selection via linear programming* (FSLP) [9]

$$\begin{aligned} \min_{w,\gamma,y,z} \quad & \left(\frac{e'y}{m} + \frac{e'z}{k}\right) + \mu e's \\ \text{subject to} \quad & -Aw + e\gamma - y \le -e, \\ & Bw - e\gamma - z \le -e, \\ & -s \le w \le s, \\ & y, \quad z \ge 0. \end{aligned} \tag{23}$$

Our FSLP formulation in (23) is slightly different from the FSV method in that FSLP is simpler to optimize and is easier to analyze in relation to the margin, which we do next. It should be noted that the normal of the separating hyperplane w in (23) has a small number of nonzero components (about 18) and a large number of 0 components (594) in our experiments. The features corresponding to the 0 components in the normal vector can be discarded, and only those with nonzero components are used.

B. Avoiding the Curse of Dimensionality

In [2], the authors did not address the *curse of dimensionality* issue. Instead, they focused on developing the FSV method to get a smooth solution, which is not explicitly connected with the margin analysis we do here. Also, their experiments used data sets in which the number of examples was much larger than the number of feature dimensions. Here we show that our FSLP method is actually related to margin maximization, which makes it possible to avoid the *curse of dimensionality* problem [14].

Consider the last term $e's$ in the objective function of (23), where s is the absolute value of the normal w due to the constraint $-s \le w \le s$. To minimize the objective function in (23) requires minimizing the term $e's$ too. Since

$$e's = \sum_i s_i = \sum_i |w_i| = \|w\|_1 \tag{24}$$

this means minimizing $\|w\|_1$, which is the 1-norm of the normal w. Because minimizing $\|w\|_1$ is equivalent to maximizing $1/\|w\|_1$, the objective function in (23) maximizes $1/\|w\|_1$.

Recall from (19) there are two bounding hyperplanes, $P1$: $w'x - \gamma = 1$ and $P2$: $w'x - \gamma = -1$. The discriminating hyperplane P is midway between these two hyperplanes, i.e., $w'x - \gamma = 0$. The distance of any point x to the hyperplane P is defined as $d(x;P) = (|w'x - \gamma|)/(\|w\|_2)$. From (19) $|w'x - \gamma| \ge 1$, so any point x that is outside the two bounding hyperplanes, $P1$ and $P2$ satisfies $d(x;P) \ge 1/\|w\|_2$.

The minimum distance between the two bounding hyperplanes is $2/||w||_2$, which is defined as the margin, similar to that used in developing SVMs [32]. The p-norm is nonincreasing monotonic for $p \in [1, \infty]$, so $||w||_1 \geq ||w||_2, \forall w \in R^n$, which is equivalent to

$$\frac{1}{||w||_1} \leq \frac{1}{||w||_2}. \tag{25}$$

The p-norm, $||w||_p$, is convex on $R^n, \forall p \in [1, \infty]$ [28]. So, by maximizing $1/||w||_1$, we approximately maximize $2/||w||_2$. As a result, the last term, $e's$, in the objective function of (23) has the effect of maximizing the margin.

Maximizing the margin can often circumvent the *curse of dimensionality* problem, as seen in SVMs, which can classify data in very high-dimensional feature spaces [32]. Our FSLP method has a similar advantage because it incorporates a feature selection process based on margin size.

In fact, when $\mu = 0$ the last term in the objective function of (23) disappears. In this case classification performance worsens in our experiments because the remaining two terms do not have the property of maximizing the margin. Actually, the item $(e'y/m + ('z/k)$ only encodes the classification error rate which usually leads to overfitting, especially in the small sample case. So, the last term, $e's$, has two effects: 1) feature selection and 2) margin maximization. These two effects are inseparable. Thus, feature selection in FSLP is based on maximizing the margin [32] instead of heuristics [30].

Because the *curse of dimensionality* problem occurs in so many computer vision tasks where the number of training examples is much smaller than the feature dimension of each example, our analysis that FSLP circumvents this problem is an important result. Further demonstration of this property is shown empirically in Section VI.

V. Pairwise Feature Selection

For a multiclass classification problem, the typical strategy is to select a fixed subset of features for each class to discriminate it from all other classes. However, this simple strategy is not optimal in discriminating between many classes. Intuitively, features useful to distinguish the letter "E" from "F" may differ from those features distinguishing "E" from "R", for example. If features are selected for all classes simultaneously, two problems can occur: 1) it is much more complex to select features to separate one class from all other classes, and 2) even if this approach works, the number of selected features will be large.

A better strategy is to select a feature subset for each pair of classes. Pairwise feature selection is useful for many multiclass classification problems in computer vision, e.g., face recognition [12]. The basic idea of pairwise feature selection is to choose the most relevant features for each pair of classes given the original high-dimensional data. A feature index table is created and stored for each pair of classes after using any feature selection method. For a c-class classification problem, there are $c(c-1)/2$ pairs. Hence, the system needs to store a table T with $c(c-1)/2$ items and each item t_{ij} is an array of indices of the features selected for the pair of classes i and j. Suppose the original feature dimension is D, the average number of features for all pairs is A, and the number of features selected by traditional techniques[2] is A'. Usually, we have the relation $A < A' < D$. Then the system stores $c(c-1)A/2$ integers to index the features. In recognition, however, there are only $(c-1)A$ features under the pairwise comparison framework or binary tournament scheme [11], [12], while one needs $(c-1)A'$ features for traditional feature selection schemes[3]. $(c-1)A < (c-1)A'$ no matter what is the size of c. On the other hand, when the one-versus-the-remaining scheme is used for recognition, it uses cA' features, and $(c-1)A < cA'$. Note that the pairwise feature selection framework cannot use the one-versus-the-remaining recognition scheme. The only requirement for pairwise feature selection is the space to store the index table T.

Note that the feature indices for classes i and j are the same as j and i, and the indices for classes i and j may partially overlap with those for i and $k, j \neq k$, but usually they are not exactly the same. After pairwise feature selection, one may count the number of occurrence of each feature for all pairs.

In this research, we let the simplified Bayes classifier, AdaBoost, and FSLP do pairwise feature selection. As shown in the experiments, different features are selected for different pairs. The top N most discriminating features are used by the simplified Bayes classifier during training. In AdaBoost, features are selected one by one according to the classification error of the weak learner in the previous step [30]. For SVMs, feature selection is not trivial [10]; while the general framework of pairwise feature selection [12] should work, the problem is how to select the features for the SVMs for each pair of classes. The ranking strategy is too simple to select appropriate features for SVMs [10]. The reason may be the difference in the optimization criterion used in SVMs versus other methods such as a Bayes classifier. Consequently, in our study all the features were used with SVMs in both training and classification.

The FSLP technique can automatically determine the number of features to select for each pair of classes, while the heuristics used for both the Bayes classifier and AdaBoost cannot determine the number of features. Feature selection is even more difficult in the case of a small number of examples because of bias [14]. In contrast, the FSLP method can select features by maximizing the margin, circumventing the *curse of dimensionality* problem.

VI. Expression Recognition Experiments

In this section we experimentally compare the four methods for FER for the case where there are a small number of training images for each expression.

A. Face Expression Database

The face expression database [18] used in our experiments contains 213 images of ten Japanese women. Each person has two to four images for each of seven expressions: neutral, happy, sad, surprise, anger, disgust and fear. Each image size

[2]Traditional techniques for feature selection choose features to discriminate among all classes, so they return the same number of features for each class.

[3]There are $(c-1)A'$ computations if the recognition scheme is pairwise, or cA' computations if the recognition is one-versus-the-remaining.

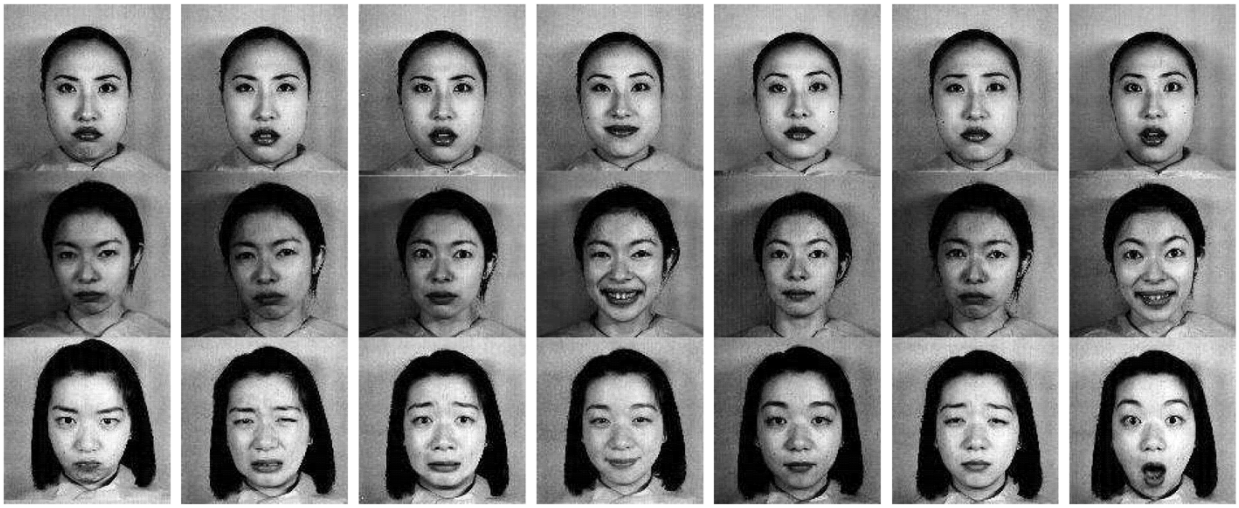


Fig. 5. Some images in the face expression database. From left to right: angry, disgust, fear, happy, neutral, sad, and surprise.

TABLE I
COMPARISON OF THE RECOGNITION ACCURACY AND THE NUMBER OF FEATURES USED BY THE BAYES CLASSIFIER WITHOUT FEATURE SELECTION (BAYES ALL), BAYES WITH PAIRWISE-GREEDY FEATURE SELECTION (BAYES FS), ADABOOST, LINEAR SVM (L-SVM), NONLINEAR SVM (NL-SVM), AND FSLP

	Bayes All	Bayes FS	AdaBoost	L-SVM	NL-SVM	FSLP
Accuracy	63.3%	71.0%	71.9%	92.4%	91.9%	91.0%
# Features	612	60	80	612	612	17.1

is 256×256 pixels. A few examples are shown in Fig. 5. For more information on the database such as image collection, data description, and human ranking, see [18]. This database was also used in [19], [38], and [37].

B. Experimental Results

Our experimental procedure used tenfold cross-validation to deal with the small sample size. That is, the database was divided randomly into ten roughly equal-sized parts, from which the data from nine parts were used for training the classifiers and the last part was used for testing. This procedure was repeated ten times so that each part was used once as the test set. This is the standard methodology used in machine learning [34].

1) Bayes Classifier and AdaBoost: The AdaBoost method [7] was adapted by Viola for solving computer vision problems such as image retrieval [30] and face detection [33], so that it uses a greedy strategy to select features in the learning phase. Greedy feature selection can also be used with a Bayes classifier by assuming feature independence and incrementally adding the most discriminating features [17]. Fig. 6 shows the recognition performance of the AdaBoost and Bayes classifiers as a function of the number of features selected. Less than 100 features are sufficient for both algorithms (the performance of the AdaBoost algorithm does not improve when more features are added). The Bayes classifier reached its best performance of 71.0% with 60 features, and the performance deteriorated slightly if more features were used. The recognition accuracy of the Bayes classifier was 63.3% (shown in Table I) when all 612 features were used. Overfitting the training data is a serious problem for the Bayes method, so feature selection is necessary. For the AdaBoost method, peak performance was 71.9% using

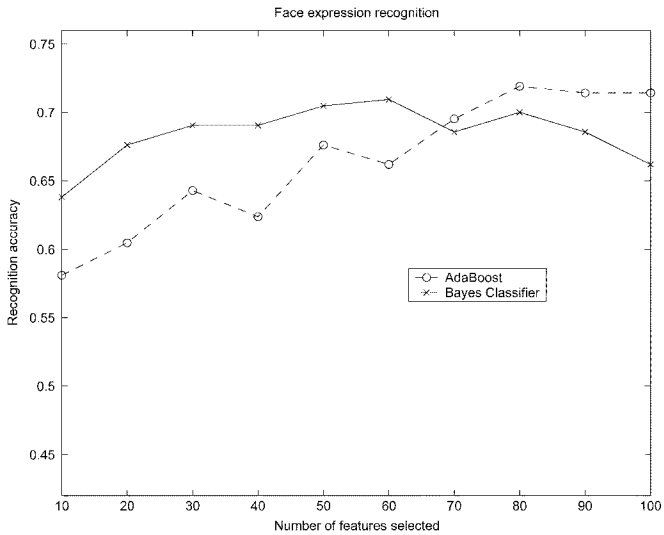


Fig. 6. Recognition accuracies of a Bayes classifier and Adaboost as a function of the number of features selected.

80 features (see Table I) for each pair of classes. As shown in Fig. 6, using more features slightly lowered recognition accuracy. In summary, both the AdaBoost and Bayes classifiers combined with a greedy feature selection strategy, selected a large number of features and their recognition accuracies were low.

2) FSLP and SVM: For our FSLP algorithm, we found that the parameter μ in (23) is best set to a small value, and we used $\mu = 0.000\,01$ in all experiments. A larger number for μ will slightly worsen the performance of FSLP but not too much. Actually, the μ value balances the error minimization and margin maximization in (23) in order to accomplish feature selection

TABLE II
PERFORMANCE OF FSLP COMPARED TO LINEAR SVM (L-SVM) AND GRBF NONLINEAR SVM (NL-SVM) USING TENFOLD CROSS-VALIDATION. THE AVERAGE NUMBER OF SELECTED FEATURES (AVE. #) FOR EACH PAIRWISE CLASSIFIER AND THE TOTAL NUMBER OF SELECTED FEATURES (TOTAL #) USED FOR ALL PAIRS ARE SHOWN IN ADDITION TO THE NUMBER OF ERRORS OUT OF 21 TEST EXAMPLES IN EACH RUN

Test	Ave. #	Total #	FSLP	L-SVM	NL-SVM
Set 1	16.8	82	3	2	1
Set 2	17.0	84	2	2	2
Set 3	17.1	90	1	1	2
Set 4	16.4	92	3	3	3
Set 5	16.0	83	1	2	2
Set 6	19.1	102	2	2	2
Set 7	16.9	85	2	2	2
Set 8	17.2	91	1	0	0
Set 9	17.5	91	2	1	2
Set 10	17.4	89	2	1	1
Ave.	17.1	88.9	1.9	1.6	1.7

and classifier training. An open problem is how to set the value of μ automatically given the training data so as to make the whole system completely automatic. To solve this 7-expression classification problem we used a simple binary tree tournament scheme with pairwise comparisons.

Experimental results using the FSLP method are shown in Table II. Feature selection was performed for each pair of classes, resulting in a total of 21 pairs for the 7-expression classification problem. The second column in Table II shows the number of selected features on average for the 21 pairwise classifiers, ranging from 16.0 to 19.1 for the ten runs. The average number of features selected over the ten runs was 17.1. Thus, a very sparse set of features was automatically selected out of the original 612 features extracted from each face image. This demonstrates that FSLP can significantly reduce the number of feature dimensions without any user interaction.

The third column in Table II shows the total number of features selected by FSLP for all 21 pairwise classifiers in each test set. Because some features are useful in discriminating between one pair, say "angry" and "happy," but not for separating another pair, say "angry" and "sad," the number of features selected for all pairs is larger than that for each pair. For instance, there were 82 features selected for 21 pairwise classifiers in Set 1. This number is still much smaller than all 612 features. On the other hand, the frequency of occurrence of the 82 features over all pairs of classes was very variable, as shown by the histogram in Fig. 7. Note that some features are used much more frequently (e.g., 15 times among 21 pairs) than others because they are discriminative in many pairs of classes. Never does a feature appear in all 21 pairs, however. Column 4 in Table II lists the number of classification errors out of 21 test examples by FSLP on each data set. The average over ten runs was 1.9.

SVMs [32] are known to give high recognition accuracy when a large number of training examples are provided. Here, we are interested in its performance in the small sample case. The classification errors of both linear and nonlinear SVMs (using all 612 features) are shown in columns 5 and 6 of Table II. For nonlinear SVM, we used the GRBF kernel and empirically set the width parameter. The maximum error of FSLP was 3 over the ten runs, which was never larger than the errors by linear SVMs and nonlinear SVMs. The average number of errors over ten runs was very similar for FSLP, linear SVM (1.6 errors) and nonlinear SVM (1.7 errors). The recognition accuracies of the three methods were 91.0%, 92.4%, and 91.9%, respectively (see Table I), which are comparable. Notice, however, that the average number of features selected by FSLP was 17.1, much less than that used by the SVMs. Furthermore, the computation time of FSLP was short in both the training and recognition phases, with run times of several minutes to train all 21 classifiers on a Linux machine with a 1.2-GHz Pentium processor using a Matlab implementation and CPLEX 6.6 for the LP.

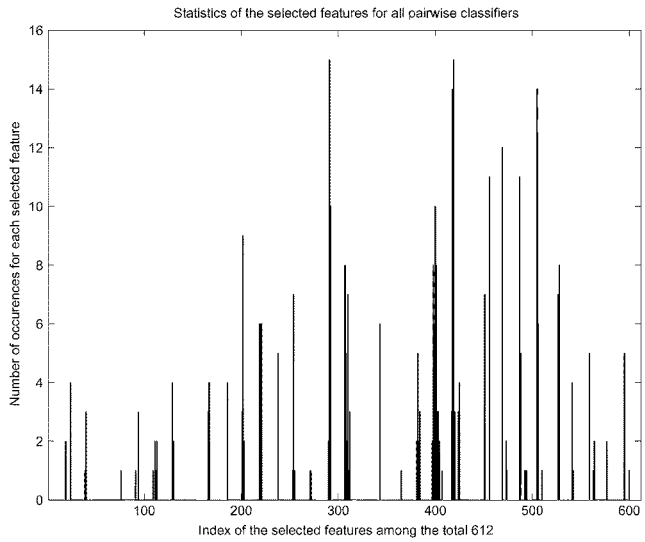


Fig. 7. Histogram of the frequency of occurrence of the 612 features used in training Set 1 for all 21 pairwise FSLP classifiers.

While the recognition accuracy of SVMs is comparable to FSLP, one major weakness of SVMs is their high computational cost, which precludes real-time applications. In addition, SVMs are formulated as a quadratic programming problem and, therefore, it is difficult to use SVMs to do feature selection directly. (Some researchers have proposed approximations to SVM for feature selection [35] by first training the SVM using the whole training set, and then computing approximations to reduce the number of features. This two-step approach cannot guarantee selection of the best feature subset, however.) Finally, SVM approximations [35] cannot determine automatically how many features to use. On the contrary, FSLP addresses all of these issues at once.

3) Comparison With Previous Approaches: Since our face expression database was used by others [38], [37], [19], we also compared recognition performance with previously published results. In [38], [37] a Neural Network was used with 90.1% recognition accuracy. When some problematic images in the database were discarded, the accuracy was 92.2%. In [19] a result of 92% using linear discriminant analysis (LDA) was reported, but they only included nine people's face images and, hence, only 193 of the 213 images were used. In conclusion,

FSLP gives comparable results to Neural Network and LDA methods, but FSLP can select a small number of features automatically.

C. Summary

Based on the experiments presented, several conclusions can be made: 1) the simplified Bayes classifier and the AdaBoost method [30] do not perform well in the small sample case; 2) SVMs can solve the recognition problem well but without feature selection capability; and 3) the FLSP algorithm can efficiently solve both the feature selection and classifier training problems simultaneously. A more general analysis of these algorithms is given in next section.

VII. Discussion

Based on the experimental results in the last section, we now present a more general analysis of learning from a small number of examples. Both FSLP and SVMs have high recognition accuracy (above 90%), while the AdaBoost [30] and the simplified Bayes classifiers gave a much lower accuracy (lower than 72%). Why such a difference? To answer this question, we think it is better to divide these algorithms into two broad categories: 1) probability distribution based methods, including the simplified Bayes classifier and AdaBoost [30] and 2) margin- based methods, such as FSLP and SVMs. Although general AdaBoost [7] is called a large margin classifier in machine learning, because there is a probability distribution in AdaBoost it is sensitive to the number of examples. Our categorization reflects the large difference in the experimental results for the small sample case.

A. Probability Distribution Based Learning

For a Bayes classifier, one needs to estimate the probability distribution $p(x \mid \omega_c)$ for each class ω_c. In the small sample case it is hard, if not impossible, for the small number of examples to "span" the underlying distribution of the unseen examples. Thus the estimated probability distribution may be biased far away from the real one. As a consequence, low accuracy can often occur. The simplified Bayes decision used with (10), (11), and (12) assumes independent and Gaussian distributions, which simplified the problem of class density estimation. However, from our experiments, the estimation is still biased, and the problem of overfitting is serious. Our pairwise feature selection framework improves the classification accuracy to some extent, but is not enough.

AdaBoost is a technique to combine weak learners that work on different distributions of the training examples. Theoretically, the only requirement for AdaBoost is that each weak learner has a performance better than 50%. However, when we look at the algorithm in detail, there is a probability distribution $w_{t,i}$ for each example $i \in [1,n]$ at each round $t \in [1,T]$. This distribution is updated for each example at each round by $w_{t+1,i} = w_{t,i}\beta_t^{e_i}$, where $\beta_t = (\epsilon_t)/(1-\epsilon_t)$ and ϵ^t is the error rate at round t. In the small sample case, when the small number of examples is not distributed in "general" positions (i.e., is biased away from the positions of the unseen examples), the estimated error rate ϵ_t is biased away from the error rate estimated when a large number of examples is available. Therefore, the distribution $w_{t,i}$ will also be biased. This bias can accumulate through each round. As a result, one cannot achieve the original goal of boosting, which focuses on the "hard" but "general" examples. In an extreme case, consider a small number of examples that are completely separated into two classes. If we add some outliers that overlap the two classes, AdaBoost will concentrate on these "hard" outliers in successive boosting rounds. As a result, the final classifier is biased by the outliers. On the contrary, margin-based SVMs and FSLP deal with outliers using slack variables.

Now look at the margin definitions of AdaBoost and SVM. In AdaBoost, the margin of example (x,y) is defined [29] by

$$\frac{y\sum_t \alpha_t h_t(x)}{\sum_t \alpha_t} \tag{26}$$

where $\alpha_t = \log(1/\beta_t) = \log(1-\epsilon_t)/(\epsilon_t)$. Hence, the margin of AdaBoost is biased in the small sample case because α_t is a biased estimation. (Note: AdaBoost may have good performance when a large number of examples are available.) In contrast, the margin in SVM is defined [32] by $(2/||w||)$, where w is the linear combination of support vectors as in (16). The margin in SVM has no relation to any probability distribution or error rate.

In our experiments, the accuracy of AdaBoost is low and similar to the simplified Bayes classifier where the decision in each feature dimension is similar to the weak learner in AdaBoost [30]. This demonstrates that AdaBoost cannot work well in the small sample case. In fact, we also observed that the performance of AdaBoost is much lower than SVM for image retrieval with relevance feedback [10], another learning problem with a small number of samples.

Based on the above discussion, we consider the Bayes and AdaBoost methods as probability distribution based learning schemes. Because probability estimation is sensitive to the number of examples or, more precisely, the distribution of the examples, these methods cannot in general solve the learning problem in the small sample case.

B. Margin-Based Learning

SVMs are well-known for margin-based discrimination. SVMs have been applied to vision problems such as face detection [23] where a large number of examples is provided for training. Our experiments demonstrate their good generalization ability in the small sample case. Maximizing the margin $(2/||w||)$ is not sensitive to the number of training examples because of its nonparametric characteristic. The reader can find other analysis of the generalization capability of SVMs in [32], and see some additional experiments for the small sample case in [6].

The FSLP algorithm maximizes the 1-norm margin $(1/||w||)$, which approximates the 2-norm margin in SVMs. We described earlier why FSLP can select features and avoid the *curse of dimensionality* problem in the small sample case (Section IV). As with SVMs, the FSLP algorithm maximizes the margin and is therefore usually insensitive to the number of examples, in contrast to the probability estimation based methods.

The FSLP method is similar to SVMs in that both maximize the margin. However, FSLP is formulated directly with the primal problem using the 1-norm, resulting in a LP problem (23). SVMs are usually transformed into a dual problem with the 2-norm [see (14)], resulting in a quadratic programming problem. The direct solution of SVMs is the dual variables α_i in (14), defining the support vectors when $\alpha_i \neq 0$. It is this difference that allows FSLP to solve the feature selection problem together with the discrimination function, whereas SVMs cannot.

In short, margin-based learning is not sensitive to the number of examples and therefore can perform well in the small sample case. The FSLP method can further deal with the feature selection problem.

VIII. Concluding Remarks

In this paper, several classification methods have been compared in the case of a small number of training examples per class. Probability distribution based learning methods such as the simplified Bayes classifier and AdaBoost cannot solve this problem. Margin-based methods such as FSLP and SVMs can accurately solve the recognition problem in the small sample case. Furthermore, FSLP can also address the feature selection problem, avoiding the *curse of dimensionality*.

Our major contributions are: 1) systematic evaluation of several popular methods in machine learning for a vision problem in the small sample case; 2) introduction of a novel algorithm called FSLP, and an analysis of how it can do feature selection together with classifier training while also circumventing the *curse of dimensionality* problem; and 3) analysis of the expected relative performance of these algorithms for learning in the small sample case, regardless of the classification task.

Acknowledgment

The authors thank O. Mangasarian and S. Wright for their help on the linear programming technique, and M. Lyons for providing the face expression database. The support of the National Science Foundation and the University of Wisconsin is also gratefully acknowledged.

References

[1] K. P. Bennett and O. L. Mangasarian, "Robust linear programming discrimination of two linearly inseparable sets," *Optimiz. Meth. Softw.*, vol. 1, pp. 23–34, 1992.

[2] P. S. Bradley and O. L. Mangasarian, "Feature selection via concave minimization and support vector machines," in *Proc. 5th Int. Conf. Machine Learning*, 1998, pp. 82–90.

[3] C. Cortes and V. Vapnik, "Support vector networks," *Mach. Learn.*, vol. 20, pp. 273–297, 1995.

[4] G. Cottrell and J. Metcalfe, "Face, gender, and emotion recognition using holons," *Adv. Neural Inform. Process. Syst. 3*, pp. 564–571, 1991.

[5] J. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by tow-dimensional visual cortical filters," *J. Opt. Soc. Amer.*, vol. A, pp. 1160–1169, 1985.

[6] R. P. W. Duin, "Classifiers in almost empty spaces," in *Proc. Int. Conf. Pattern Recognit.*, vol. 2, 2000, pp. 1–7.

[7] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of online learning and an application to boosting," *J. Comp. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[8] K. Fukunage, *Introduction to Statistical Pattern Recognition*, 2nd ed. New York: Academic, 1991.

[9] G. Guo and C. R. Dyer, "Simultaneous feature selection and classifier training via linear programming: A case study for face expression recognition," in *Proc. CVPR*, vol. I, Jun. 2003, pp. 346–352.

[10] G. Guo, A. K. Jain, W. Y. Ma, and H. J. Zhang, "Learning similarity measure for natural image retrieval with relevance feedback," in *Proc. CVPR*, vol. 1, 2001, pp. 731–736.

[11] G. Guo, S. Z. Li, and K. L. Chan, "Face recognition by support vector machines," *Image Vis. Comput.*, vol. 19, no. 9–10, pp. 631–638, 2001.

[12] G. Guo, H. J. Zhang, and S. Z. Li, "Pairwise face recognition," in *Proc. ICCV*, vol. 2, 2001, pp. 282–287.

[13] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognit.*, vol. 16, no. 12, pp. 1167–1186, 1991.

[14] A. Jain and D. Zongker, "Feature selection: Evaluation, application, and small sample performance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 2, pp. 153–158, Feb. 1997.

[15] M. Lades, J. C. Vorbruggen, J. Buhmann, J. Lange, C. von der Malsburg, R. P. Wurtz, and W. Konen, "Distortion invariant object recognition in the dynamic link architecture," *IEEE Trans. Comput.*, vol. 42, no. 3, pp. 300–311, Mar. 1993.

[16] A. Lanitis, C. Taylor, and T. Cootes, "Automatic interpretation and coding of face images using flexible models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 743–756, Jul. 1997.

[17] C. Liu and H. Wechsler, "Probablistic reasoning models for face recognition," *Proc. CVPR*, pp. 827–832, 1998.

[18] M. J. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding facial expressions with gabor wavelets," in *Proc. 3rd IEEE Int. Conf. Automatic Face and Gesture Recognition*, 1998, pp. 200–205.

[19] M. J. Lyons, J. Budynek, and S. Akamatsu, "Automatic classification of single facial images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 12, pp. 1357–1362, Dec. 1999.

[20] O. L. Mangasarian, "Linear and nonlinear separation of patterns by linear programming," *Oper. Res.*, vol. 13, pp. 444–452, 1965.

[21] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 8, pp. 837–842, Aug. 1996.

[22] A. M. Martinez, "Recognizing imprecisely localized, partially occluded, and expression variant faces from a single sample per class," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 6, pp. 748–763, Jun. 2002.

[23] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: An application to face detection," in *Proc. Computer Vision and Pattern Recognition Conf.*, 1997, pp. 130–136.

[24] C. Padgett and G. Cottrell, "Identifying emotion in static images," in *Proc. 2nd Joint Symp. Neural Computation*, vol. 5, 1997, pp. 91–101.

[25] M. Pantie and L. J. M. Rothkrantz, "Automatic analysis of facial expressions: The state of the art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 12, pp. 1424–1445, Dec. 2000.

[26] P. Pudil, J. Novovicova, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognit. Lett.*, vol. 15, pp. 1119–1125, 1994.

[27] A. Rahardja, A. Sowmya, and W. Wilson, "A neural network aproach to component versus holistic recognition of facial expressions in images," in *SPIE Proc., Intelligent Robots and Computer Vision X: Algorithms and Techniques*, vol. 1607, 1991, pp. 62–70.

[28] R. T. Rockafellar, *Convex Analysis*. Princeton, NJ: Princeton Univ. Press, 1970.

[29] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the margin: A new explanation for the effectiveness of voting mehtods," *Annal. Statist.*, vol. 26, no. 5, pp. 1651–1686, 1998.

[30] K. Tieu and P. Viola, "Boosting image retrieval," in *Proc. Computer Vision and Pattern Recognition Conf.*, vol. 1, 2000, pp. 228–235.

[31] A. Vailaya, "Semantic Classification in Image Database," Ph.D. dissertation, Michigan State Univ., East Lansing, 2000.

[32] V. N. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[33] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. CVPR*, vol. 1, 2001, pp. 511–518.

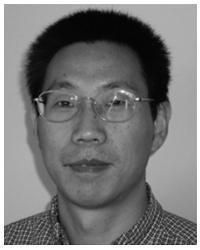
[34] S. M. Weiss and C. K. Kulikowski, *Computer Systems that Learn*. San Mateo, CA: Morgan Kaufmann, 1991.

[35] J. Weston, "Feature selection for SVMs," *Adv. Neural Inform. Process. Syst.*, vol. 13, 2000.

[36] L. Wiskott, J.-M. Fellous, N. Kruger, and C. von der Malsburg, "Face recognition by elastic bunch graph matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 775–779, Jul. 1997.

[37] Z. Zhang, "Feature-based facial expression recognition: Sensitivity analysis and experiments with a multi-layer perceptron," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 13, no. 6, pp. 893–911, 1999.

[38] Z. Zhang, M. Lyons, M. Schuster, and S. Akamatsu, "Comparison between geometry-based and Gabor-wavelets-based facial expression recognition using multi-lyaer perceptron," in *Proc. 3rd IEEE Int. Conf. Automatic Face and Gesture Recognition*, 1998, pp. 454–459.



Guodong Guo received the B.E. degree in automation from Tsinghua University, Beijing, China, in 1991 and the Ph.D. degree in pattern recognition and intelligent control from the Institute of Automation, Chinese Academy of Sciences, in 1998.

He is currently a Research Assistant in the Computer Sciences Department, University of Wisconsin-Madison (UW-Madison). He was a Visiting Researcher at INRIA, Sophia Antipolis, France, in 1997, at Ritsumeikan University, Japan, in 1998, and Nanyang Technological University, Singapore, during 2000–2001. He also worked at Microsoft Research China for one year before he moved to UW-Madison. His research interests include computer vision, machine learning, multimedia information retrieval, face recognition, and image analysis. He has published about 30 technical papers in these areas.

Charles R. Dyer (S'75–M'79–SM'85–F'98) received the B.S. degree in mathematical sciences from Stanford University, Stanford, CA, in 1973, the M.S. degree in computer science from UCLA in 1974, and the Ph.D. degree in computer science from the University of Maryland, College Park, in 1979.

From 1979 to 1982, he was an Assistant Professor in the Department of Electrical Engineering and Computer Science, University of Illinois at Chicago. He has been on the Faculty at the University of Wisconsin–Madison since 1982, where he is currently a Professor of Computer Sciences and Professor of Biostatistics and Medical Informatics. He has worked in the field of computer vision for 30 years and has published over 130 technical papers. His research has focused mainly on shape and image representations for computer vision, with contributions on topics such as quadtrees, aspect graphs, pyramid algorithms, image-based modeling and rendering, object recognition, and motion analysis.

Dr. Dyer was General Co-Chair of the 2003 Computer Vision and Pattern Recognition Conference. He was awarded the Visiting Fellowship at Exeter College, University of Oxford, Oxford, U.K., for 1999–2000. He is a Fellow of the IAPR.