7

Multi-View Face Recognition with Min-Max Modular Support Vector Machines

Zhi-Gang Fan and Bao-Liang Lu[1]
Department of Computer Science and Engineering, Shanghai Jiao Tong University
Shanghai
China

1. Introduction

As a result of statistical learning theory, support vector machines (SVMs)[23] are effective classifiers for the classification problems. SVMs have been successfully applied to various pattern classification problems, such as handwritten digit recognition, text categorization and face detection, due to their powerful learning ability and good generalization ability. However, SVMs require to solve a quadratic optimization problem and need training time that are at least quadratic to the number of training samples. Therefore, many large-scale problems by using traditional SVMs are too hard to be solved. To overcome this difficulty, Lu and colleagues have proposed a min-max modular support vector machine (M^3-SVM) and part-versus-part task decomposition method [16]. A very important advantage of M^3-SVMs over traditional SVMs is that a two-class problem can be further decomposed into a series of two-class subproblems.

The M^3-network model [15] has been applied successfully to many real-world applications such as part-of-speech tagging [17], single-trial EEG signal classification [18], prediction of protein subcellular multi-locations [26], face recognition [2, 13] and text categorization [14]. The basic idea behind M^3-network is the "divide and conquer" strategy. The task decomposition scheme of M^3-network is based on class relations, and the instances in the same class can be further decomposed randomly [15], according to parallel hyperplanes [24], or prior knowledge [13]. The learning procedure of each subproblems is independent, and therefore parallel learning can be implemented easily. The combination strategy follows two principles, the minimization principle and the maximization principle [15].

We explore the use of M^3-SVMs in multi-view face recognition. Multi-view face recognition is a more challenging task than frontal view face recognition. Face recognition techniques have been developed over the past few decades. But many of those existing face recognition techniques, such as Eigenfaces and Fisher-faces [22, 1], are only effective for frontal view faces. The difficulties of multi-view face recognition is obvious because of the complicated nonlinear manifolds existing in the data space. Using M^3-SVMs, we can decompose the

[1] To whom correspondence should be addressed. This work was supported in part by the National Natural Science Foundation of China under the grants NSFC 60375022 and NSFC 60473040, and The Microsoft Laboratory for Intelligent Computing and Intelligent Systems of Shanghai Jiao Tong University.

Source: Face Recognition, Book edited by: Kresimir Delac and Mislav Grgic, ISBN 978-3-902613-03-5, pp.558, I-Tech, Vienna, Austria, June 2007

whole complicated problem of multi-view face recognition into several relatively simpler two-class sub-problems. Every individual two-class sub-problem becomes less complicated than the original problem and it can be solved effectively. In addition, we use a SVM based discriminative feature selection (SVM-DFS) method [3] for feature selection in multi-view face recognition.

2. Part-Versus-Part Task Decomposition

For human beings, the only way to solve a complex problem is to divide it into smaller, more manageable subproblems. Breaking up a problem helps human beings deal with complex issues involved in its solution [18]. This "divide-and- conquer" strategy is also helpful to neural networks and machine learning approaches for dealing with complex learning problems. Our goal in this Section is to introduce a part-versus-part task decomposition method for training multi-class SVMs.

Let $\mathcal{T}$ be the given training data set for a ***K***-class classification problem,

$$\mathcal{T} = \{(X_l, \hat{Y}_l)\}_{l=1}^{L}, \tag{1}$$

where $X_l \in \mathcal{X} \subset \mathbf{R}^n$ is the input vector, $\mathcal{X}$ is the set of training inputs, $\hat{Y}_l \in \mathcal{Y} \subset \mathrm{R}^K$ is the desired output, $\mathcal{Y}$ is the set of desired outputs, and ***L*** is the total number of training data. We have suggested that a ***K***-class problem defined by (1) can be divided into $K(K–1) = 2$ two-class subproblems [15], each of which is given by

$$\begin{aligned} \mathcal{T}_{ij} = \{(X_l^{(i)}, +1)\}_{l=1}^{L_i} \cup \{(X_l^{(j)}, -1)\}_{l=1}^{L_j} \\ \text{for } i = 1, \cdots, K \text{ and } j = i+1, \cdots, K \end{aligned} \tag{2}$$

where $X_l^{(i)} \in \mathcal{X}_i$ and $X_l^{(j)} \in \mathcal{X}_j$ are the training inputs belonging to class $\mathcal{C}_i$ and class $\mathcal{C}_j$, respectively, $\mathcal{X}_i$ is the set of training inputs belonging to class $\mathcal{C}_i$, L_i denotes the number of data in $\mathcal{X}_i$, $\cup_{i=1}^{K} \mathcal{X}_i = \mathcal{X}$, and $\sum_{i=1}^{K} L_i = L$.

In this Chapter, the training data in a two-class subproblem are called *positive* training data if their desired outputs are +1. Otherwise, they are called *negative* training data. The two-class subproblems defined by (2)
are called *pair-wise classification* in the machine learning literature [5,11]. We would like to emphasize that decomposition of a ***K***-class problem into $\boldsymbol{K}(\boldsymbol{K}–1)/2$ two-class subproblems defined by (2) is unique for a given training data set because of the uniqueness of $\mathcal{X}$ for i=1,...,***K***.

Although the two-class subproblems defined by (2) are smaller than the original ***K***-class problem, this partition may not be adequate for parallel computation and fast learning. To speed up learning, all the large and imbalanced two-class subproblems should be further divided into relatively smaller and more balanced two-class subproblems.

Assume that $\mathcal{X}_i$ is partitioned into N_i subsets in the form

$$\begin{aligned} \mathcal{X}_{ij} = \{X_l^{(ij)}\}_{l=1}^{L_i^{(j)}} \\ \text{for } j = 1, \cdots, N_i \text{ and } i = 1, \cdots, K, \end{aligned} \tag{3}$$

where $1 \le N_i \le L_i$ and $\cup_{j=1}^{N_i} \mathcal{X}_{ij} = \mathcal{X}_i$.

Various methods can be used for partitioning $\mathcal{X}_i$ into N_i subsets [15]. A simple and straightforward approach is to divide $\mathcal{X}_i$ randomly. The subsets $\mathcal{X}_{ij}$ might be disjoint or joint. Without loss of generality and for simplicity of description, we assume throughout this Chapter that the random decomposition method is used and the subsets $\mathcal{X}_{ij}$ are disjoint from each other, i.e., $\mathcal{X}_{ij} \cap \mathcal{X}_{ik} = \phi$ for i =1,...,K, j and k=1,...,N_i, and $j \neq k$.

In practical applications of SVMs, an appropriate value of N_i might depend on two main factors, such as the number of training data belonging to each class and the available computational power. In the simulations presented in this Chapter, we randomly divide $\mathcal{X}_i$ into N_i subsets $\mathcal{X}_{ij}$, which are roughly the same in size. The number of subsets N_i for class $\mathcal{C}_i$ is determined according to the following rule:

$$b_i = \begin{cases} d + x_1 \text{ if } fmod\ (\frac{2L_i}{\rho}) \leq \gamma \text{ and } 2L_i > \rho \\ \lceil \frac{2L_i}{\rho} \rceil \text{ otherwise} \end{cases} \tag{4}$$

where ρ is the desired number of training data fort wo-class subproblems, γ is a threshold parameter ($0<\gamma<1$) for fine-tuning the number of subsets, $\lfloor z \rfloor$ denotes the largest integer less than or equal to z, $\lceil z \rceil$ denotes the smallest integer larger than or equal to z, the function of $f\ mod(z_1/z_2)$ is employed to produce the decimal part of z_1/z_2, and z_1 and z_2 are two positive integers, respectively.

After partitioning $\mathcal{X}_i$ into N_i subsets, every two-class subproblem $\mathcal{T}_{ij}$ defined by (2) can be further divided into $N_i \times N_j$ relatively smaller and more balanced two-class subproblems as follows:

$$\begin{aligned} \mathcal{T}_{ij}^{(u,v)} = \{(X_l^{(iu)},\ +1)\}_{l=1}^{L_i^{(u)}} \cup \{(X_l^{(jv)},\ -1)\}_{l=1}^{L_j^{(v)}} \\ \text{for } u = 1,\ \cdots,\ N_i,\ v = 1,\ \cdots,\ N_j, \\ i = 1,\ \cdots,\ K,\ \text{and } j = i+1,\ \cdots,\ K \end{aligned} \tag{5}$$

where $\mathcal{X}_l{}^{(iu)} \in \mathcal{X}_{iu}$ and $\mathcal{X}_l{}^{(jv)} \in \mathcal{X}_{jv}$ are the training inputs belonging to class $\mathcal{C}_i$ and class $\mathcal{C}_j$, respectively, $\sum_{u=1}^{N_i} L_i^{(u)} = L_i$ and $\sum_{v=1}^{N_j} L_j^{(v)} = L_j$. It should be noted that all the two-class subproblems have the same number of input dimensions as the original ***K***-class problem. Comparing the two-class subproblems defined by (5) with the two-class subproblems obtained by the pairwise-classification approach, we can see that each of the two-class subproblems defined by (5) containsonly apart of data of each class. Hence, the decomposition method is called *part-versus-part* method [16].

According to the above discussion, the part-versus-part task decomposition method can be described as Table 1.

After task decomposition, each of the two-class subproblems can be treated as a completely independent, non-communicating problem in the learning phase. Therefore, all the two-class subproblems can be efficiently learned in a massively parallel way.

From (2) and (5), we see that a ***K***-class problem can be divided into

$$\sum_{i=1}^{K-1} \sum_{j=i+1}^{K} N_i \times N_j \tag{6}$$

two-class subproblems. The number of training data for each of the two-class subproblems is about

$$\lceil L_i/N_i \rceil + \lceil L_j/N_j \rceil \tag{7}$$

Since $\lceil L_i/N_i \rceil + \lceil L_j/N_j \rceil$ is independent of the number of classes ***K***, the size of each of the two-class subproblems is much smaller than the original ***K***-class problem for reasonable N_i and N_j.

Step 1: Set the values of ρ and γ.
Step 2: Divide a ***K***-class problem $\mathcal{T}$ into $\binom{K}{2}$ two-class subproblems $\mathcal{T}_{ij}$ using (2).
Step 3: If the sizes of all τij are less than ρ, then stop the procedure here. Otherwise, continue with the following steps.
Step 4: Determine the number of training input subsets N_i for i=1,...,K using (4).
Step 5: Divide the training input set $\mathcal{X}_i$ into N_i subsets $\mathcal{X}_{ij}$ using (3).
Step 6: Divide the two-class subproblem $\mathcal{T}_{ij}$ into $N_i \times N_j$ relatively smaller and simpler two class subproblems $\mathcal{T}_{ij}^{(u,v)}$ using (5).

Table 1. The part-versus-part task decomposition method

3. Min-Max Modular Support Vector Machine

Before using M³-SVMs, for a ***K***-class problem, we should divide the ***K***-class problem into $K(K-1)/2$ two-class sub-problems according to one-against-one strategy or divide a ***K***-class problem into ***K*** two-class subproblems according to one-against-all strategy. In this work, we use one-against-one strategy. The work procedure of M³-SVMs consists of three steps: task decomposition, SVMs training and module combination. First, every two-class problem is decomposed into relatively smaller two-class problems. Then, every smaller two-class SVM is trained. At last, all of the modules are integrated into a M³-SVM to obtain the final solutions to the original problem.

3.1 Support Vector Machine

Support vector machine is a machine learning technique that is well-founded in statistical learning theory. The SVM algorithm formulates the training problem as a problem that finds, among all possible separating hyperplanes, one hyperplane that maximizes the distance between the closest elements of the two classes. In practice, this is determined through solving a quadratic programming problem. SVMs have a general form of decision function for an input x as:

$$f(x) = \operatorname{sign}\left(\sum_{\text{support vectors}} y_i \alpha_i K(x_i, x) - b \right) \tag{8}$$

where α_i are Lagrange parameters obtained in the optimization step, y_i are class labels, and $K(\cdot,\cdot)$ is the kernel function. The kernel function can be various types.
The linear kernel function is $K(x,y)=x \cdot y$; the radial-basis function kernel function is $K(x,y) = \exp\left(-\frac{1}{2\sigma^2}\|x-y\|^2\right)$ and the polynomial kernel function is $K(x,y)=(x \cdot y+1)^n$.

3.2 Module Combination

After training, all the individual SVMs are integrated into aM³-SVM with the MIN unit and the MAX unit according to the following two combination principles: the minimization principle and the maximization principle [15,16].

Minimization Principle: Suppose a two-classproblem $\mathcal{B}$ were divided into P relatively smallert wo-class subproblems, $\mathcal{B}_i$ for i=1,...,P, and also suppose that all the two-class subproblems have the same positive training data and different negative training data. If the P two-class subproblems are learned by the corresponding P individual SVMs, M_i for i=1,...,P, then the combination of the P trained SVMs with a MIN unit will produce the correct output for all the training inputs in $\mathcal{B}$, where the function of the MIN unit is to find a minimum value from its multiple inputs. The transfer function of the MIN unit is given by

$$q(x) = \min_{i=1}^{P} \mathrm{M}_i(x) \tag{9}$$

where x denotes the input variable.

Maximization Principle: Suppose a two-classproblem $\mathcal{B}$ were divided into P relatively smaller two-class subproblems, $\mathcal{B}_i$ for i=1,...,P, and also suppose that all the two-class subproblems have the same negative training data and different positive training data. If the P two-class subproblems are learned by the corresponding P individua lSVMs, M_i for i=1,...,P, then the combination of the P trained SVMs with a MAX unit will produce the correct output for all the training input in $\mathcal{B}$, where the function of the MAX unit is to find a maximum value from its multiple inputs. The transfer function of the MAX unit is given by

$$q(x) = \max_{i=1}^{P} \mathrm{M}_i(x) \tag{10}$$

For example, a two-class problems defined by (2) is further divided into $N+ \times N-$ relatively smaller two-class subproblems. After learning all of these two-class subproblems with SVMs, the trained $N+ \times N-$ individual SVM modules are integrated into a M³-SVM with $N+$ MIN units and one MAX unit as follows:

$$\mathrm{M}_{ij}^{(u)}(x) = \min_{v=1}^{N^-} \mathrm{M}_{ij}^{(u,v)}(x) \text{ for } u = 1, 2, \cdots, N^+ \tag{11}$$

and

$$\mathrm{M}_{ij}(x) = \max_{u=1}^{N^+} \mathrm{M}_{ij}^{(u)}(x) \tag{12}$$

where $\mathrm{M}_{ij}^{(u,v)}(x)$ denotes the transfer function of the trained SVM corresponding to the two-class subproblem $\mathcal{T}_{ij}^{(u,v)}$, and $\mathrm{M}_{ij}^{(u)}(x)$ denotes the transfer function of a combination of N− SVMs integrated by the MIN unit. Figure 1 illustrates the structure of a M³-SVM.

Suppose that a 1-out-of-K scheme were used for output representation. Let Y denote the actual output vector of the M³-SVM for a K-class classification problem, and let $g(x)$ denote the transfer function of the entire M³-SVM. We may then write

$$Y = g(x) = [g_1(x), \cdots, g_K(x)]^T \tag{13}$$

According to the minimization and maximization principles, the $\binom{K}{2}$ SVMs, M_{ij} (x) for i=1,...,K and j=i+1,...,K, and the corresponding $\binom{K}{2}$ inversions M_{rs}(x) for r=2,...,K and s=1,...,r − 1, are integrated as

$$g_i(x) = \min\left[\min_{j=i+1}^{K} \mathrm{M}_{ij}(x), \min_{r=1}^{i-1} \overline{\mathrm{M}_{ri}(x)}\right] \tag{14}$$

where g_i (x) for i=1,...,K denotes the discriminant function, which discriminates the patterns of class $\mathcal{C}_i$ from those of th eremaining classes, and the term $\overline{\mathrm{M}_{ri}(x)}$ denotes the inversion of M_{ri} (x).

It is easy to implement $\overline{\mathrm{M}_{ri}(x)}$ with M_{ri} (x) and an INV unit. The function of the INV unit is to invert its single input; the transfer function of the INV unit is given by

$$q = \alpha + \beta - p \tag{15}$$

where a, β, p ,and q are the upper and lower limits of input value input, and output, respectively. For example, a and β are set to +1 and -1, respectively, for support vector classifiers in the simulations below.

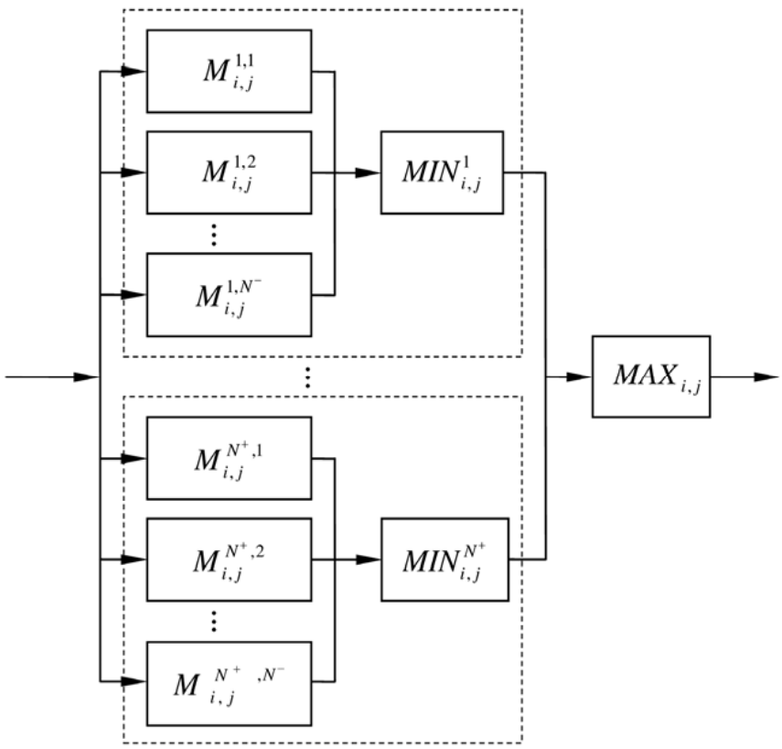

Figure 1. Structure of a M3-SVM consisting of N+ × N− individual SVMs, N+ MIN units, and one MAX unit

The relationship among $M_{rs}(x)$, $\overline{M_{sr}(x)}$, and the INV unit can be expressed as

$$\mathrm{M}_{rs}(x) = \overline{\mathrm{M}_{sr}(x)} = \mathrm{INV}(\mathrm{M}_{sr}(x)) \quad \text{for } s = 1, \cdots, K-1;\ r = s+1, \cdots, K \tag{16}$$

Similarly, the discriminant function g_i (x) of the Min-Max SVM, which consists of $\sum_{i=1}^{K-1}\sum_{j=i+1}^{K} N_i \times N_j$ network modules, and the corresponding $\binom{K}{2}$ inversions can be expressed as

$$g_i(x) = \min\left[\min_{j=i+1}^{K}\left[\max_{k=1}^{N_i}\left[\min_{l=1}^{N_j} \mathrm{M}_{ij}^{(k,l)}(x)\right]\right] \min_{r=1}^{i-1}\overline{\left[\max_{k=1}^{N_r}\left[\min_{l=1}^{N_i} \mathrm{M}_{ri}^{(k,l)}(x)\right]\right]}\right] \tag{17}$$

where the term $\overline{\max_{k=1}^{N_r}[\min_{l=1}^{N_i} \mathrm{M}_{ri}^{(k,l)}(x)]}$ denotes the inversion of $\max_{k=1}^{N_r}[\min_{l=1}^{N_i} \mathrm{M}_{ri}^{(k,l)}(x)]$. It should be noted that only the inversions of network modules $M_{i\,j}$ (*x*) are used for constructing the M³-SVMs, and there are no inversions for SVMs $\mathrm{M}_{ij}^{(u,v)}(x)$.

Summarizing the discussion mentioned above, the module combination procedure can be described as Table 2.

Step 1: If no SVMs $\mathrm{M}_{ij}^{(u,v)}(x)$ exist, go to Step 3. Otherwise, perform the following steps.
Step 2: Integrate $N_i \times N_j$ SVMs $\mathrm{M}_{ij}^{(u,v)}(x)$ for u = 1,.. , N_i, v=1,..., N_j, i =1,..., K, and j=i+1,...,K, into a module $M_{i\,j}$ (*x*) with N_i MIN units and one MAX unit according to (11) and (12).
Step 3: Integrate ***K*** (***K***– 1)/2 modules and the corresponding ***K*** (***K***– 1)/2 inversions with ***K*** MIN units according to (14).

Table 2. The module combination procedure

From the module combination procedure above, we see that individual trained SVMs can be simply integrated into a M³-SVM with MIN, MAX and/or INV units. Since the module combination procedure is completely independent of both the structure of individual trained SVMs and their performance, we can easily replace any trained SVMs with desired ones to achieve better generalization performance. In contrast to the task decomposition procedure mentioned earlier, the module combination procedure proceeds in a bottom-up manner. The smaller trained SVMs arei ntegrated into larger modules first, and then the larger modules arei ntegrated into a M³-SVM.

After finishing module combination, the solutions to the original ***K***-class problem can be obtained from the outputs of the entire M³-SVM as follows:

$$\mathcal{C} = \arg\max_{i}\{g_i(x)\} \text{ for } i = 1, \cdots, K \tag{18}$$

where $\mathcal{C}$ is the class that the M³-SVM assigns to the input *x*.

Once the size of each of the SVMs is fixed, the space complexity of the entire M³-SVM is determined according to (14) and (17). Table 3 shows the number of individual SVM modules and integrating units required to construct a M³-SVM for a ***K***-class problem.

4. Discriminative Feature Selection

We use a SVM-based discriminative feature selection (SVM-DFS) [3] method for multi-view face recognition in this study.

Name	#elements
SVMs	$2\sum_{i=1}^{K-1}\sum_{j=i+1}^{K} N_i \times N_j$
MIN	$K + 2\sum_{i=1}^{K-1}\sum_{j=i+1}^{K} N_i \lceil \frac{N_j - 1}{N_j} \rceil$
MAX	$2\sum_{i=1}^{K-1}(K-i)\lceil \frac{N_i - 1}{N_i} \rceil$
INV	$K(K-1)/2$

Table 3. Number of SVM modules and integrating units required to build the M³-SVM for a ***K***-class problem (***K***>2)

4.1 Feature Selection in Binary Classification

In the linear case of binary classification, the decision function equation (8) can be reformed as

$$f(x) = \text{sign } (w \cdot x - b) \tag{19}$$

where w obtained from

$$w = \sum_{\text{support vectors}} y_i \alpha_i x_i \tag{20}$$

The inner product of weight vector w=(w_1,w_2,...,w_n) and input vector x=(x_1,x_2,...,x_n) determines the value of $f(x)$. Intuitively,the input features in a subset of (x_1,x_2,...,x_n) that are weighted by the largest absolute value subset of (w_1,w_2,...,w_n) influence most the classification decision. If the classifier performs well, the input features subset with the largest weights should correspond to the most informative features. Therefore, the weights $|w_i|$ of the linear decision function can be used as feature ranking criterion [7] [8] [25] [3] [10] [4] [20] [9] [19]. According to the feature ranking criterion, we can select the most

discriminative features for the binary classification task. However, this way for feature ranking is a greedy method and we should look for more evidences for feature selection.

Support vectors can be used as evidence for feature ranking [3] [10] [4], because support vectors can be used to count for different distributions of the features in the training data. Assume the distance between the optimal hyperplane and the support vectors is Δ, the optimal hyperplane can be viewed as a kind of Δ-margin separating hyperplane which is located in the center of margin (−Δ, Δ). According to [23], the set of Δ-margin separating hyperplanes has the VC dimension h bounded by the inequality

$$h \leq \min\left(\left[\frac{R^2}{\Delta^2}\right], n\right) + 1 \tag{21}$$

where $\boldsymbol{R}$ is the radius of a sphere which can bound the training vectors x∈$\boldsymbol{X}$. Inequality (21) points out the relationship between margin Δ and VC dimension: a larger Δ means a smaller VC dimension. Therefore, in order to obtain high generalization ability, we should still maintain margin large after feature selection. However, because the dimensionality of original input space has been reduced after featur eselection, the margin is usually to shrink and what we can do is trying our best to make the shrink small to some extent. Therefore, in feature selection process, we should preferentially select the features which make more contribution to maintaining the margin large. This is another evidence for feature ranking. To realize this idea, a coefficient c_k is introduced,

$$c_k = \left| \frac{1}{l_+} \sum_{i \in SV_+} x_{i,k} - \frac{1}{l_-} \sum_{j \in SV_-} x_{j,k} \right| \tag{22}$$

where SV_+ denotes the support vectors belong to positive samples, SV_- denotes the support vectors belong to negative samples, l_+ denotes the number of SV_+, l_-denotes the number of SV_-, and $x_{i,k}$ denotes the kth feature of support vector i in input space $\boldsymbol{R}^n$. The larger c_k indicates that the kth feature of input space can make more contribution to maintaining the margin large. Therefore, c_k can assist $|w_k|$ for feature ranking. The solution is that, combining the two evidences, we can order the features by ranking $c_k\ |w_k|$.

In the nonlinear case of binary classification, a cost function $\boldsymbol{J}$ is computed on training samples for feature ranking. $\boldsymbol{DJ}(i)$ denotes the change in the cost function $\boldsymbol{J}$ caused by removing a given feature or, equivalently, by bringing its weight to zero. $\boldsymbol{DJ}(i)$ can be used as feature ranking criterion. In [7], $\boldsymbol{DJ}(i)$ is computed by expanding $\boldsymbol{J}$ in Taylor series to second order. At the optimum of $\boldsymbol{J}$, the first order term can be neglected, yielding

$$DJ(i) = \frac{1}{2}\frac{\partial^2 J}{\partial w_i^2}(Dw_i)^2 \tag{23}$$

where the change in weight Dw_i corresponds to removing feature i.

For the nonlinear SVMs with the nonlinear decision function $\boldsymbol{f}(x)$, the cost function $\boldsymbol{J}$ being minimized is

$$J = \frac{1}{2}\alpha^T H \alpha - \alpha^T v \tag{24}$$

where $\boldsymbol{H}$ is the matrix with elements $y_h\ y_k\ \boldsymbol{K}\ (x_h,x_k)$, α is Lagrange parameter vector $\alpha=(\alpha_1,\alpha_2,...,\alpha_n)$, and v is a n dimensional vector of ones [7]. To compute the change in cost function caused by removing input component i, one leaves the α's unchanged and one recomputes matrix $\boldsymbol{H}$. This corresponds to computing $\boldsymbol{K}(x_h\ (-i),\ x_k\ (-i))$, yielding matrix $H(-i)$, where the notation $(-i)$ means that component i has been removed. Thus, the feature ranking criterion for nonlinear SVMs is

$$DJ(i) = \frac{1}{2}\left(\alpha^T H\alpha - \alpha^T H(-i)\alpha\right) \tag{25}$$

Computation of $\boldsymbol{DJ}(i)$ is a little more expensive than that in the linear case. However, the change in matrix $\boldsymbol{H}$ must be computed for support vectors only, which makes it affordable for small numbers of support vectors.

For the convenience of representation, in both linear and nonlinear cases of binary classification, we denote feature ranking criterion as r_i for the ith feature in the input space $\boldsymbol{R}^n$. In linear case of binary classification, r_i is

$$r_i = c_i|w_i| \tag{26}$$

In nonlinear case of binary classification, r_i is

$$r_i = \frac{1}{2}\left(\alpha^T H\alpha - \alpha^T H(-i)\alpha\right) \tag{27}$$

Using feature ranking criterion r_i, we can select most discriminative features for binary classification task.

4.2 Feature Selection in Multi-class Classification

In the case of multi-class classification, we use one-versus-all method for multi-class SVMs. Multi-class classification problem is much more difficult than the binary one especially when the data are of high dimensionality and the sample size is small. The classification accuracy appears to degrade very rapidly as the number of classes increases [12]. Therefore, feature selection in multi-class classification is more challenging than that in binary case. We should be more careful when extending feature selection from binary case to multi-class case. Using the statistical relationship between feature ranking and the multiple sub-models of multi-class SVMs, we propose the SVM-DFS method for features election.

One-versus-all multi-class SVMs constructs $\boldsymbol{K}$ decision functions where $\boldsymbol{K}$ is the number of classes. The jth decision function $\boldsymbol{f}_j(x)$ is constructed with all of the examples in the jth class with positive labels, and all other examples with negative labels. The $\boldsymbol{f}_j(x)$ is a binary classification sub-model for discriminating the jth class from the all other classes. When $\boldsymbol{f}_j(x)$ has the maximum value among all the sub-models, $\boldsymbol{f}_j(x)$ has determined the classification result that the jth class is true. The $r_{i\ j}$, calculated from $\boldsymbol{f}_j(x)$, denotes the feature ranking criterion of the ith feature according to the binary classification sub-model $\boldsymbol{f}_j(x)$. There are sure event $\boldsymbol{E}$ and impossible event $\boldsymbol{Ø}$ in probability theory. Let ω_j denote the event that the jth class is true. According to probability theory, events $\omega_1,\omega_2,...,\omega_k$ constitute a partition of the sample space

$$E = \omega_1 \cup \omega_2 \cup \ldots \cup \omega_k \tag{28}$$

and

$$\emptyset = \omega_i \cap \omega_j, \quad i \neq j \tag{29}$$

$\boldsymbol{P}$ (ω_j) is the prior probability that the jth class is true. Define a random event S_i as "the ith feature is selected as discriminative feature". Let $\boldsymbol{P}(S_i \mid \omega_j)$ denote the conditional probability of S_i given that ω_j occurred. When event ω_j occur, the jth binary classification sub-model $\boldsymbol{f}_j(x)$ has the maximum value among all the sub-models and it is just uniquely effective for determining the final classification result

$$P(\omega_j | f_j(x) \text{ is effective}) = P(f_j(x) \text{ is effective} | \omega_j) = 1 \tag{30}$$

on the premise that the $\boldsymbol{f}_j(x)$ is correct. Under the condition that the jth binary classification sub-model $\boldsymbol{f}_j(x)$ is effective, we can calculate $\boldsymbol{P}(S_i \mid \omega_j)$ through the feature ranking criterion r_{ij}

$$P(S_i|\omega_j) = P(S_i|f_j(x) \text{ is effective}) = \frac{r_{ij}}{\sum_{t=1}^{n} r_{tj}} \tag{31}$$

According to the theorem on the total probability, $\boldsymbol{P}(S_i)$ can be calculated through $\boldsymbol{P}(S_i \mid \omega_j)$ and $\boldsymbol{P}(\omega_j)$

$$P(S_i) = \sum_{j=1}^{K} P(S_i|\omega_j) P(\omega_j) \tag{32}$$

Then, $\boldsymbol{P}(S_i)$ can be used as feature ranking criterion for the whole multi-class classification problem. The solution is that we can order the features by ranking $\boldsymbol{P}(S_i)$ and select the features which have larger value of $\boldsymbol{P}(S_i)$. In Table 4, we present an outline of the SVM-DFS algorithm.

In the algorithm, T and M_t are two user defined constants. T is the number of the iteration steps. Usually, T should not be too small. M_t is the number of the features to be selected in the t iteration step. M_t can be evaluated by retraining the SVM classifiers with the M_t selected features. M_t should be set to such a value that the margin Δ_i of each retrained SVM sub-model $\boldsymbol{f}_i(x)$ is large enough

$$\Delta_i = \frac{1}{\|w^{(i)}\|} \tag{33}$$

where $w^{(i)}$ denotes the weight vector of sub-model $\boldsymbol{f}_i(x)$.According to [23],

$$\|w^{(i)}\|^2 = \sum_{\text{support vectors}} \alpha_j^{(i)} \tag{34}$$

where $\alpha_j^{(i)}$ denotes Lagrange parameter of sub-model $\boldsymbol{f}_i(x)$. Define a coefficient L:

$$L = \sum_{i=1}^{k} P(\omega_i) \left(\sum_{\text{support vectors}} \alpha_j^{(i)} \right) \tag{35}$$

- Input:
  Training examples
  $$X_0 = \{x_1, x_2, \ldots x_l\}^T$$
- Initialize:
  Indices for selected features: s=[1,2,...n]
  Train the SVM classifier using samples X_0
- For t=1,...,T:
  1. Compute the ranking criteria $\boldsymbol{P}(S_i)$ according to the trained SVMs
  2. Order the features by decreasing $P(S_i)$, select the top M_t features, and eliminate the other features
  3. Updates by eliminating the indices which not belong to the selected features
  4. Restrict training examples to selected feature indices
     ```
     X=X0(:,s)
     ```
  5. Train the SVM classifier using samples X
- Outputs:
  The small set of critical features and the final SVM classifier

Table 4. The outline of the SVM-DFS algorithm

We can use coefficient L to evaluate M_t. M_t should be set to such a value that the value of L is small enough. After the M_t discriminative features have been selected through SVM-DFS, the SVM models have to be retrained using the training data.

5. Experiments

We use the UMIST database [6], am ulti-view face database consisting of 575 gray-scale images of 20 subjects. Each of the subjects covers a wide range of poses from profile to frontal views. Figure 2 depicts some sample images of a subject in the UMIST database. This is a classification problem of 20 classes. The overall database is partitioned into two subsets: the training set and test set. The training set is composed of 240 images of 20 persons: 12 images per person are carefully chosen according to face poses. The remaining 335 images are used to form the test set. All input images are of size 112×92. We have used SVM-DFS discriminative feature selection method to reduce the dimensionality of feature space. All of the experiments were performed on a 3.0 GHz Pentium 4 PC with 1.0 GB RAM.

After nonlinear dimensionality reduction [21], the distribution of face poses is shown in Figgure 3. From Figgure 3, we can see that the distribution of faces varies based on face poses. Following the observation from Figgure 3, we partition the set of training inputs for each class into four subsets by using the part-versus-part task decomposition strategy. As a result, the original 20-class classification problem has been decomposed into 3040 two-class subproblems. First, the origial 20-class classification problem has been decomposed into (20*(20-1))/2=190 two-class subproblems. Second, each two-class subproblem has been decomposed to 4*4=16 two-class subproblems. Therefore, the original problem has been decomposed into (20*(20-1))/2*4*4=3040 two-class subproblems. Every individual subproblem becomes less complicated than the original problem and it can be solved more effectively.

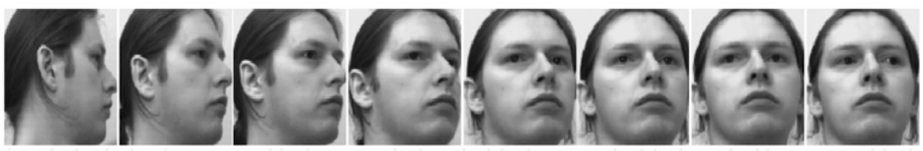

Figure 2. Some face samples of one subject from the UMIST face database

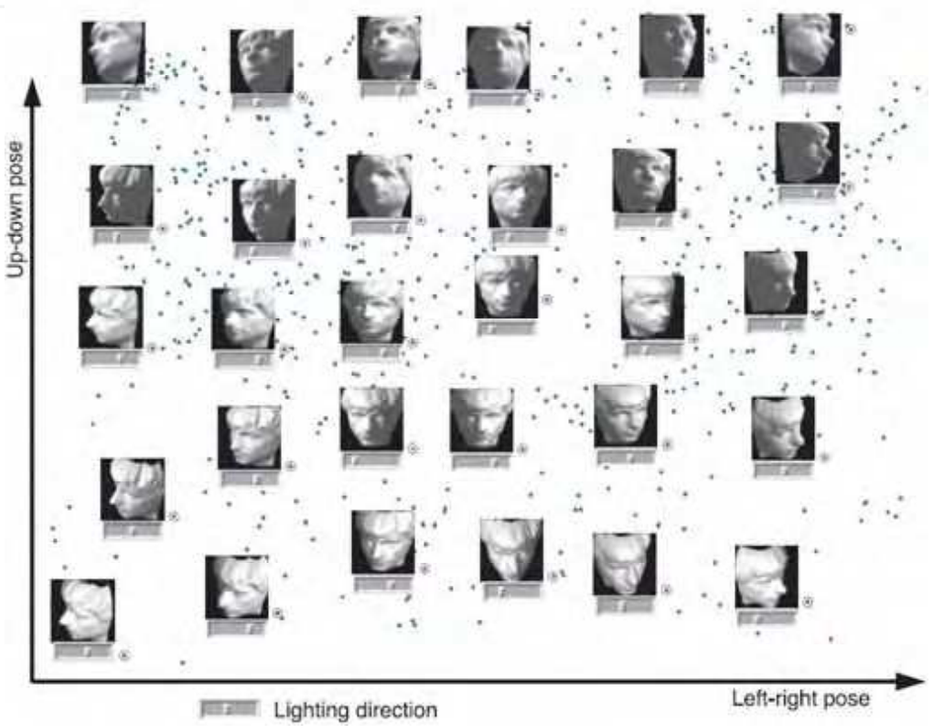

Figure 3. Distribution of face poses is shown after nonlinear dimensionality reduction (From Tenenbaum et al.[21])

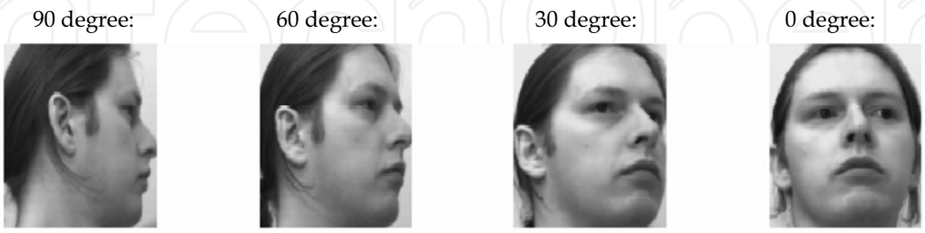

Figure 4. Training face images for each class are divided into 4 subsets according to face poses

Methods	No. features	σ	Training time (s)		Test time (s)	Correct rate (%)
			Parallel	Serial		
SVMs (rbf kernel)	300	30	0.862	13.588	1.522	92.8358
	200	25	0.748	12.654	0.976	92.2388
	150	25	0.703	11.865	0.757	90.1493
	100	20	0.685	11.269	0.478	82.3881
M³-SVMs(rbfkernel)	300	20	0.531	15.273	1.647	93.1343
	200	15	0.447	13.413	1.215	92.5373
	150	10	0.386	12.587	0.873	91.3433
	100	10	0.359	12.165	0.526	83.8806

Table 5. Test results on UMIST face database

To evaluate the effectiveness of the proposed method, the multi-view face recognition problem was learned by both M³-SVMs and standard SVMs. The one-versus-all method is used for training the standard SVMs. A radial-basis function kernel for SVMs is used, the parameter C=10000, and σ is set to the optimal values. The experimental results are shown in Table 5. From Table 5, we can see that M³-SVMs can obtain better generalization performance than the standard SVMs when the original problem is decomposed into 3040 two-class subproblems, and meanwhile the training time can be reduced in a parallel way. The parallel training is to train all the sub-modules at the same time in parallel. And the serial training is to train all the individual modules one-by-one in serial. In parallel training way, M³-SVMs can make the training speed faster comparing to the standard SVMs. The results in Table 5 also indicate that even though in low feature space after discriminative feature selection, M³-SVMs are still more accurate than the standard SVMs.

6. Conclusions

We have applied the min-max modular support vector machine and the part-versus-part task decomposition method to dealing with multi-view face recognition problems. We have demonstrated that face pose information can be easily incorporated into the procedure of dividing a multi-view face recognition problem into a series of relatively easier two-class subproblems. We have performed some experiments on the UMIST database and compared with the standard support vector machines. The experimental results indicate that the min-max modular support vector machine can improve the accuracy of multi-view face recognition and reduce the training time. As a future work, we will perform experiments on large-scale face databases with various face poses. We believe that the min-max modular support vector machine with incorporating pose information into task decomposition will have more advantages over traditional support vector machines in both training time and recognition accuracy when a more number of training samples are available.

7. References

Belhumeur, P., Hespanda, J., Kiregeman, D. Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, (1997) 711-720. [1]

Fan, Z. G., Lu, B. L. (2005). Multi-view Face Recognition with Min-Max Modular SVMs, *Proc. ICNC '05, Lecture Notes in Computer Science*, vol.3611, pp.396-399. [2]

Fan, Z. G., Lu, B. L. Fast Recognition of Multi-View Faces with Feature Selection, *10th IEEE International Conference on Computer Vision* (ICCV'05), vol. 1, pp. 76-81, 2005. [3]

Fan, Z. G., Wang K. A., Lu, B. L. Feature Selection for Fast Image Classification with Support Vector Machines, *Proc. ICONIP 2004* , LNCS 3316, pp. 1026-1031, 2004. [4]

Friedman, J. H., Another approach to polychotomous classification, *Technical Report*, (ftp://stat.stanford.edu/pub/friedman/poly.ps.Z), Stanford University, 1996. [5]

Graham, D.B., Allinson, N.M. Characterizing virtual eigensignatures for general purpose face recognition. In: Face Recognition: From Theory to Applications, NATO ASI Series F, *Computer and Systems Sciences*, vol. 163, (1998) 446-456. [6]

Guyon, I., Weston, J., Barnhill, S., Vapnik, V. Gene Selection for Cancer Classification using Support Vector Machines, *Machine Learning*, Vol. 46, pp. 389-422, 2002. [7]

Guyon, I., Elisseeff, A. An introduction to variable and feature selection, *Journal of Machine Learning*, Vol. 3, pp. 1157-1182, 2003. [8]

Guyon, I., Gunn, S. R., Ben-Hur, A., Dror, G. Result Analysis of the NIPS 2003 Feature Selection Challenge, *NIPS 2004*, 2004. [9]

Heisele, B., Serre, T., Prentice, S., Poggio, T. Hierarchical classification and feature reduction for fast face detection with support vector machine, *Pattern Recognition*, Vol. 36, pp. 2007-2017, 2003. [10]

Krebel, U., Pairwise classification and support vector machines, In B. SchÄolkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernal Methods {Support Vector Learning)*, pp. 255-268, Cambridge, MA, 1999, MIT Press. [11]

Li, T., Zhang, C., Ogihara, M. A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression, *Bioinformatics*, Vol. 20, No. 15, pp. 2429-2437, 2004. [12]

Lian, H. C., Lu, B. L., Takikawa, E, Hosoi, S. (2005). Gender Recognition Using a Min-Max Modular Support Vector Machine, *Proc. ICNC'05, Lecture Notes in Computer Science*, vol.3611, pp.438-441. [13]

Liu, F. Y., Wu, K., Zhao, H., Lu, B. L. (2005). Fast Text Categorization with Min-Max Modular Support Vector Machines, *IJCNN '05*, vol.1, pp570-575. [14]

Lu, B.L., Ito, M. Task decomposition and module combination based on class relations: a modular neural network for pattern classification. *IEEE Transactions on Neural Networks,* vol.10, (1999) 1244 -1256 [15]

Lu, B.L., Wang, K.A., Utiyama, M., Isahara, H. A part-versus-part method for massively parallel training of support vector machines. In: *Proceedings of IJCNN '04*, Budapast, July 25-29(2004) [16]

Lu, B. L., Ma, Q., Ichikawa, M., Isahara, H. (2003). E±cient Part-of-Speech Tagging with a Min-Max Modular Neural Network Model, *Applied Intelligence*, vol.19, pp.65-81. [17]

Lu, B. L., Shin, J., Ichikawa, M. (2004). Massively Parallel Classification of Single-Trial EEG Signals Using a Min-Max Modular Neural Network, *IEEE Trans. Biomedical Engineering*, vol. 51, pp. 551-558. [18]

Mao, K. Z. Feature Subset Selection for Support Vector Machines Through Discriminative Function Pruning Analysis, *IEEE Trans. Systems, Man, and Cybernetics*, vol. 34, no. 1, 2004. [19]

Mladenic, D., Brank, J., Grobelnik, M., Milic-Frayling, N. Feature selection using linear classifier weights: interaction with classification models", *Proceedings of the 27th annual international ACM SIGIR conference*, Vol. 1, pp. 234-241, 2004. [20]

Tenenbaum, J. B., Silva, V. De, Langford, J. C., (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, Vol. 290, No. 5500, pp. 2319- 2323. [21]

Turk M., Pentland, A. Eigenfaces for Recognition. *Journal of Cognitive Neuro-science*, vol. 3, no. 1, (1991) 71-86 [22]

Vapnik, V. N. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York (2000) [23]

Wang, K. A., Zhao, H., Lu, B. L. (2005). Task Decomposition Using Geometric Relation for Min-Max Modular SVMs, *ISNN 2005, Lecture Notes in Computer Science*, vol.3 496, pp. 887-892. [24]

Weston, J., Elisseeff, A., Schoelkopf, B., Tipping, M. Use of the zero norm with linear models and kernel methods, *Journal of Machine Learning*, Vol. 3, pp. 1439- 1461, 2003. [25]

Yang, Y., Lu, B. L. (2006). Prediction of Protein Subcellular Multi-Locations with a Min-Max Modular Support Vector Machined, in *Proceedings of Third International Symposium on Neural Networks* (ISNN 2006). [26]

Face Recognition

Edited by Kresimir Delac and Mislav Grgic

ISBN 978-3-902613-03-5

Hard cover, 558 pages

Publisher I-Tech Education and Publishing

Published online 01, July, 2007

Published in print edition July, 2007

This book will serve as a handbook for students, researchers and practitioners in the area of automatic (computer) face recognition and inspire some future research ideas by identifying potential research directions. The book consists of 28 chapters, each focusing on a certain aspect of the problem. Within every chapter the reader will be given an overview of background information on the subject at hand and in many cases a description of the authors' original proposed solution. The chapters in this book are sorted alphabetically, according to the first author's surname. They should give the reader a general idea where the current research efforts are heading, both within the face recognition area itself and in interdisciplinary approaches.

InTech Europe

University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China

Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen