AN APPROACH TO LARGE MARGIN DESIGN OF PROTOTYPE-BASED PATTERN CLASSIFIERS

Tingting HE, Yu HU, Qiang HUO

Department of Computer Science, The University of Hong Kong, Hong Kong, China
(E-mails: tthe@cs.hku.hk, yuhu@cs.hku.hk, qhuo@cs.hku.hk)

ABSTRACT

In this paper, we propose a maximum separation margin (MSM) training method for multiple-prototype(MP)-based pattern classifiers in which a sample separation margin defined as the distance from the training sample to the classification boundary can be calculated precisely. Similar to support vector machine (SVM) methodology, MSM training is formulated as a multicriteria optimization problem which aims at maximizing the separation margin and minimizing the empirical error rate on training data simultaneously. By making certain relaxation assumptions, MSM training can be reformulated as a semidefinite programming (SDP) problem that can be solved efficiently by some standard optimization algorithms designed for SDP. Evaluation experiments are conducted on the task of the recognition of most confusable Kanji character pairs identified from popular Nakayosi and Kuchibue handwritten Japanese character databases. It is observed that the MSM-trained MP-based classifier achieves a similar character recognition accuracy as that of the state-of-the-art SVM-based classifier, yet requires much fewer classifier parameters.

Index Terms— large margin, pattern classification, support vector machine, machine learning, semidefinite programming.

1. INTRODUCTION

Discriminative training (DT) has been extensively studied over the past two decades and been demonstrated quite effective in improving the performance over the traditional maximum likelihood (ML) training in many pattern recognition applications. The DT methods try to minimize the empirical error rate on training set such that a good testing performance can be expected when the testing and training conditions match well. However, when there exist mismatches between training and testing conditions, the DT methods may suffer a serious overtraining problem. Therefore, how to achieve a good generalization ability becomes an important issue in classifier design. Large margin classifiers, such as support vector machines (SVMs) (e.g., [11]), were invented to address the above concern by maximizing a so-called *separation margin* which reflects somehow a "distance" from a training sample to the classification boundary. Consequently, many algorithms that explicitly or implicitly exploit the concept of margins have been developed in the past decade for pattern classifier design. More recently, researchers in automatic speech recognition (ASR) field have started to explore the potential of this methodology for estimating classifier parameters based on models such as Gaussian mixture models (GMMs) [10] and continuous-density HMMs (CDHMMs) (e.g., [5, 7, 6, 13]). Promising results have been achieved on several ASR benchmark databases. However, the so-called *margin* of a training sample used in the above works is defined with respect to (w.r.t.) a discriminant function as proposed by [1] (hereinafter referred to as *discriminant margin*) rather than w.r.t. classification boundary as defined in the original SVM formulation [11].

In this paper, we propose a maximum separation margin (MSM) training method for multiple-prototype(MP)-based pattern classifiers in which the sample separation margin defined as the distance from the sample to the classification boundary can be calculated precisely. Similar to training SVMs, MSM training is formulated as a multicriteria optimization problem which aims at maximizing the separation margin and minimizing the empirical error rate on training data simultaneously. Different from *discriminant margin* based approaches, the training objective function based on the separation margin in our case is bounded by definition. By making certain relaxation assumptions, MSM training can be reformulated as a semidefinite programming (SDP) [12] problem that can be solved efficiently by some standard optimization algorithms [3] designed for SDP. No heuristic tunning is needed in optimization because SDP is known as a convex optimization problem in which a global optimum can be found.

The rest of the paper is organized as follows. We present the formulation of MSM training problem in Section 2, and its solution in Section 3. Experimental results are reported in Section 4. Finally, we conclude the paper in Section 5.

2. FORMULATION OF MSM TRAINING PROBLEM

Suppose there are M classes $\{C_i\}_{i=1}^{M}$, each being modeled by K_i prototypes, $\lambda_i = \{m_{ik}\}_{k=1}^{K_i}$, where each prototype m_{ik} is a D-dimensional vector. We use $\Lambda = \{\lambda_i\}_{i=1}^{M}$ to denote the set of prototype parameters. In the pattern classification step, the feature vector X is compared with each of the M class models and a discriminant function using Euclidean distance as dissimilarity measure is computed for each class C_i as follows:

$$g_i(X;\Lambda) = -\min_k \|X - m_{ik}\|^2. \tag{1}$$

The class that gives the maximum discriminant score is considered to be the recognized class, i.e.,

$$X \in C_i \quad if \quad i = \arg\max_j g_j(X;\Lambda). \tag{2}$$

Suppose we are given a set of training samples $\mathcal{X} = \{\mathcal{X}_i\}_{i=1}^{M}$, where $\mathcal{X}_i = \{X_i^{(j)}\}_{j=1}^{N_i}$ represents the set of N_i training samples

This research was supported by grants from the RGC of the Hong Kong SAR (Project Numbers HKU7145/03E and HKU7136/05E).

for class C_i with $X_i^{(j)}$ being the j-th training sample. Let's use N_{tr} to denote the total number of training samples, i.e., $N_{tr} = \sum_{i=1}^{M} N_i$. For each training sample $X_i^{(j)}$, the corresponding prototype index of the true class, the indices of the most confused class and its corresponding prototype are determined as follows:

$$\begin{aligned}\hat{k} &= \arg\min_{k} \|X_i^{(j)} - m_{ik}\|^2, \\ q &= \arg\max_{r, r\neq i} g_r(X_i^{(j)}; \Lambda), \\ \bar{k} &= \arg\min_{k} \|X_i^{(j)} - m_{qk}\|^2. \end{aligned} \tag{3}$$

In the following, we will use $g_i(X)$ instead of $g_i(X;\Lambda)$ for convenience if no confusion will be caused according to the context of relevant discussions.

2.1. Definition of Sample Separation Margin

For each training sample $X_i^{(j)}$, the sample margin $\rho(X_i^{(j)})$ is defined as the distance to the class boundary determined by prototypes $m_{i\hat{k}}$ and $m_{q\bar{k}}$, which can be derived as,

$$\rho(X_i^{(j)}) = \frac{g_i(X_i^{(j)}) - g_q(X_i^{(j)})}{2\|m_{i\hat{k}} - m_{q\bar{k}}\|}. \tag{4}$$

The derivation is outlined as follows.

Given two prototypes of two competing classes, $m_{i\hat{k}}$ and $m_{q\bar{k}}$, the class boundary separating them is a hyperplane with a normal vector $(m_{i\hat{k}} - m_{q\bar{k}})$, which passes through the point, $(m_{i\hat{k}} + m_{q\bar{k}})/2$. According to the triangle theorem in geometry,

$$\begin{aligned} g_i(X_i^{(j)}) - g_q(X_i^{(j)}) &= -\|X_i^{(j)} - m_{i\hat{k}}\|^2 + \|X_i^{(j)} - m_{q\bar{k}}\|^2 \\ &= d_1^2 - d_2^2 = r_1^2 - r_2^2 \\ &= (r_1 + r_2)(r_1 - r_2), \end{aligned} \tag{5}$$

where d_1, d_2, r_1, r_2 are the relevant distances as shown in Fig. 1. When $\rho(X_i^j) \leq \frac{\|m_{i\hat{k}} - m_{q\bar{k}}\|}{2}$, which corresponds to scenario (a) in Fig. 1, it is obvious that

$$r_1 = \frac{\|m_{i\hat{k}} - m_{q\bar{k}}\|}{2} + \rho(X_i^{(j)}), r_2 = \frac{\|m_{i\hat{k}} - m_{q\bar{k}}\|}{2} - \rho(X_i^{(j)}). \tag{6}$$

Substituting Eq. (6) to Eq. (5), we can get

$$g_i(X_i^{(j)}) - g_q(X_i^{(j)}) = 2\rho(X_i^{(j)}) \cdot \|m_{i\hat{k}} - m_{q\bar{k}}\|.$$

Same result can be derived for the case corresponding to scenario (b) in Fig. 1, where $\rho(X_i^{(j)}) \geq \frac{\|m_{i\hat{k}} - m_{q\bar{k}}\|}{2}$.

Actually, the above sample margin $\rho(X_i^{(j)})$, reflecting the relative position to the classification boundary, is a normalized version of the *discriminant margin*, $g_i(X_i^{(j)}) - g_q(X_i^{(j)})$.

2.2. Maximum Separation Margin Training

When the training samples are separable, MSM training is to estimate the classifier parameters Λ by maximizing the minimum sample margin as follows:

$$\hat{\Lambda} = \arg\max_{\Lambda} \min \rho(X_i^{(j)}). \tag{7}$$

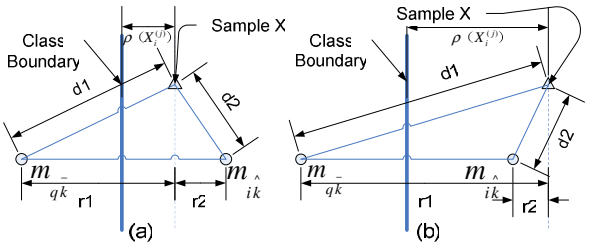

Fig. 1. Definition of sample margin in a prototype-based classifier: (a) $\rho(X_i^{(j)}) \leq \frac{\|m_{i\hat{k}} - m_{q\bar{k}}\|}{2}$ and (b) $\rho(X_i^{(j)}) \geq \frac{\|m_{i\hat{k}} - m_{q\bar{k}}\|}{2}$.

If we introduce a new variable ρ as the lower bound of sample margins for all the training samples, then the above $maxmin$ problem can be converted to an equivalent constrained optimization problem as follows:

$$\hat{\Lambda} = \arg\max_{\Lambda} \rho \quad \text{subject to} \quad \forall X_i^{(j)} \in \mathcal{X} \quad \rho(X_i^{(j)}) \geq \rho. \tag{8}$$

In general case where some training samples are inseparable, it is important to make a good balance between maximizing the classifier margin and minimizing the empirical error rate on the training set. So a nonnegative slack variable $\xi_i^{(j)}$ is introduced for each training sample as in soft SVM to satisfy $\rho(X_i^{(j)}) + \xi_i^{(j)} \geq \rho$ for the outlier samples and inseparable cases. A "weighted sum" approach is used here to integrate the above two criteria into a unified one as follows:

$$\text{Minimize } -\rho + \gamma_1 \sum_i \sum_j \ell(\xi_i^{(j)}) \tag{9}$$

Subject to

$$\rho(X_i^{(j)}) + \xi_i^{(j)} \geq \rho \tag{10}$$

$$\xi_i^{(j)} \geq 0, \tag{11}$$

where the loss function $\ell(\xi_i^{(j)})$ is defined as

$$\ell(\xi_i^{(j)}) = \begin{cases} 1 & \xi_i^{(j)} > 0 \\ 0 & \xi_i^{(j)} = 0 \end{cases} \tag{12}$$

and γ_1 is the penalty weight for slack variables.

Although the above formulation of MSM training problem is appealing, the discontinuous loss function and the min operator embedded in $g_i(X_i^{(j)})$ in Eq. (1) make the above optimization problem difficult to solve. Therefore, in this study, we propose the following iterative procedure for MSM training:

Step 1: *Initialization*

First, LBG clustering algorithm [8] is used to obtain the initial prototypes, m_{ik}^0, for each class.

Step 2: *Updating* m_{ik}

Given the m_{ik}^0's, the true prototype index $\hat{k}$ for each training sample can be identified. Then, m_{ik}'s are updated by solving the fol-

lowing relaxed optimization problem:

Minimize

$$-\tilde{\rho} + \gamma_1 \sum_i \sum_j \tilde{\xi}_i^{(j)} + \gamma_2 \sum_{i,n,i<n} \sum_{k,l} \|m_{ik} - m_{nl}\|^2 \quad (13)$$

Subject to

$$\forall n \neq i \forall l - \|X_i^{(j)} - m_{i\hat{k}}\|^2 + \|X_i^{(j)} - m_{nl}\|^2 + \tilde{\xi}_i^{(j)} \geq \tilde{\rho} \quad (14)$$

$$\sum_i \sum_k \|m_{ik} - m_{ik}^0\|^2 \leq r^2 \quad (15)$$

$$\tilde{\xi}_i^{(j)} \geq 0. \quad (16)$$

The constraint in Eq. (15) is imposed to make sure the "feasible region" is restricted as the neighborhood area with the radius r around the current prototypes $\{m_{ik}^0\}$ so that the prototype index $\hat{k}$ is meaningful.

Step 3: *Repeat* **Step 2** *until convergence*

Set $m_{ik}^0 = m_{ik}$ and repeat **Step 2** until a pre-specified criterion is satisfied, e.g., a fixed number of iterations, N_{total}.

3. SDP FORMULATION AND SOLUTION

Generally speaking, the optimization problem in Step 2 of the above MSM training procedure is a non-convex optimization problem. However, it can be reformulated into a standard SDP problem if further relaxation assumptions are made.

Let's arrange all $L = \sum_{i=1}^{M} K_i$ prototype vectors into a $D \times L$ matrix as $U = [m_{11} \cdots m_{ij} \cdots m_{MK_M}]$ and all slack variables into a $N_{tr} \times N_{tr}$ diagonal matrix $\Xi = Diag\{\tilde{\xi}_1^{(1)}, ..., \tilde{\xi}_i^{(j)}, ..., \tilde{\xi}_M^{(N_M)}\}$. The dissimilarity measure between $X_i^{(j)}$ and any prototype m_{ck} can be calculated as

$$\begin{aligned} A_{ck}(X_i^{(j)}) &= -\|X_i^{(j)} - m_{ck}\|^2 \\ &= -(X_i^{(j)}; e_{ck})^T [I_D; U]^T [I_D; U] (X_i^{(j)}; e_{ck}), \end{aligned} \quad (17)$$

where $e_{ck} \in \Re^L$ is the vector of all zeros except a -1 at the $\hat{f}(c,k)$-th position ($\hat{f}(c,k) = \Sigma_{i=1}^{c-1} K_i + k$); I_D is a $D \times D$ identity matrix; and $(X_i^{(j)}; e_{ck}) \in \Re^{D+L}$ is the concatenated vector of $X_i^{(j)}$ and e_{ck}. Similarly, the Euclidean distance between any two prototypes and the Euclidean distance between any prototype and its initial setting can be calculated as

$$\|m_{ck} - m_{nl}\|^2 = e_{ck,nl}^T U^T U e_{ck,nl}, \quad (18)$$

$$\|m_{ck} - m_{ck}^0\|^2 = (m_{ck}^0; e_{ck})^T [I_D; U]^T [I_D; U] (m_{ck}^0; e_{ck}), \quad (19)$$

where $e_{ck,nl} \in \Re^L$ is a vector with 1 at the $\hat{f}(c,k)$-th position, -1 at the $\hat{f}(n,l)$-th position and zeros elsewhere.

Since $Y = U^T U$ is non-convex, by following the practice in [12], we relax the constraint $Y = U^T U$ to $Y \succeq U^T U$, where $Y \succeq U^T U$ means that $Y - U^T U$ is semidefinite, i.e., $Y - U^T U \succeq 0$. It is well-known that the condition $Y \succeq U^T U$ is equivalent to

$$Z = \begin{pmatrix} I_D & U \\ U^T & Y \end{pmatrix} \succeq 0.$$

Therefore, we can reformulate the above relaxed problem as a standard SDP problem, namely, to find a symmetric matrix $Z \in \Re^{(D+L)\times(D+L)}$, a diagonal matrix $\Xi \in \Re^{N_{tr} \times N_{tr}}$, and a variable $\tilde{\rho}$, which

$$\begin{aligned} &\text{minimize} \quad -\tilde{\rho} + (\gamma_1 I) \cdot \Xi + (\gamma_2 B) \cdot Z, \quad (20) \\ &\text{subject to} \\ &\forall X_i^{(j)} \forall n \neq i \forall l \quad \tilde{\rho} - \hat{A}_{i\hat{k},nl}(X_i^{(j)}) \cdot Z - S_{ij} \cdot \Xi \leq 0, \\ &Q \cdot Z \leq r^2, \\ &Z_{1:D,1:D} = I_D, \\ &Z \succeq 0, \ \Xi \succeq 0, \ \tilde{\rho} \geq 0, \end{aligned}$$

where $B, Q, \hat{A}_{i\hat{k},nl}(X_i^{(j)})$ are coefficient matrices required in SDP formulation and defined as

$$\begin{aligned} B &= \sum_{i<j} \sum_{k,l} (0; e_{ik,jl})(0; e_{ik,jl})^T, \\ Q &= \sum_i \sum_k (m_{ik}^0; e_{ik})(m_{ik}^0; e_{ik})^T, \\ \hat{A}_{i\hat{k},nl}(X_i^{(j)}) &= A_{i\hat{k}}(X_i^{(j)}) - A_{nl}(X_i^{(j)}), \end{aligned} \quad (21)$$

and S_{ij} is a $N_{tr} \times N_{tr}$ diagonal matrix with all zeros except 1 at the $\bar{f}(i,j)$-th diagonal element (Note that $\bar{f}(i,j) = \Sigma_{k=1}^{i-1} N_k + j$).

The above SDP problem can then be solved by using the DSDP software [3].

4. EXPERIMENTS AND RESULTS

4.1. Experimental Setup

In order to evaluate the effectiveness of the proposed MSM training method, a series of comparative experiments are conducted on the task of the recognition of most confusable Kanji character pairs identified from popular Nakayosi and Kuchibue handwritten Japanese character databases [9]. Table 1 lists those five pairs of confusable Kanji characters with their SJIS codes and the corresponding number of samples in training, development and test sets of each character group. It is noted that samples for each Kanji character from Nakayosi database are used for training, while samples from Kuchibue database are partitioned randomly into two sets corresponding to development set and test set, respectively.

For feature analysis, a 512-dimensional raw feature vector is extracted from each character sample by using the approach proposed in [2]. A new 64-dimensional feature vector is then obtained via LDA (Linear Discriminant Analysis) transformation which is estimated by using training samples from 2965 level-1 Kanji characters, and is used for constructing different character classifiers in the following experiments.

For single-prototype(SP)-based classifiers, the initial value of the prototype is simply the mean of the training feature vectors of each character class. For MP-based classifiers, LBG clustering algorithm [8] is used first to obtain the initial prototypes for each character class, followed by MSM training.

In SDP-based MSM training, the control parameter r is set as $r = 0.4\sqrt{L}$, and $N_{total} = 1$. For SP-based binary classifiers, we set $\gamma_2 = 0$ in Eq. (20), and introduce one more constraint $B \cdot Z \leq d^2$ instead for the ease of implementation, where $d = ||m_{11}^0 - m_{21}^0||$. For all other cases, the above standard SDP problem in Eq. (20) is solved with $\gamma_2 = 0.005$ and γ_1 being tuned on development set from 0.02 to 0.12 with a step size of 0.005 for binary classifiers, and from 0.005 to 0.025 with a step size of 0.001 for the multiclass classifier, respectively.

Table 1. Five most confusable Kanji character pairs identified from Nakayosi and Kuchibue databases, and the corresponding number of samples in training, development and test sets.

Group	SJIS Code	Training Set	Dev. Set	Test Set
A	sjisx8f5e/89fa	326	79	160
B	sjisx88be/8c49	326	78	160
C	sjisx8cf3/8cf2	326	119	240
D	sjisx96a2/9696	326	199	400
E	sjisx928f/88a3	326	78	160

Table 2. A comparison of character recognition rates (in %) of LBG-trained MP-based classifiers with 1 or 2 prototypes, SVM-based classifiers with linear or RBF kernel functions, and MSM-trained MP-based classifiers with 1 or 2 prototypes on three two-class classification tasks (A, B, C) and one ten-class classification task (*All*, including A, B, C, D, E).

	LBG		SVM		MSM	
	1	2	Linear	RBF	1	2
A	80.00	84.38	96.25	95.63	96.25	96.88
B	82.50	83.13	93.50	93.75	93.75	94.38
C	86.67	90.83	95.00	94.58	95.00	95.42
All	83.13	82.77	90.63	90.89	90.71	90.80

For comparison, the popular toolkit LIBSVM [4] is used to train two types of SVM-based classifiers for the same tasks. One is based on linear SVM, the other is based on RBF (radial basis function) SVM. Two control parameters, namely the penalty weight C and the hyperparameter γ in RBF kernel function (see [4] for details), are tuned on the development set of each recognition task to obtain the best-performing classifiers. For linear SVM, the penalty weight C is scheduled to change from 0.05 to 1 with a step size of 0.05. For RBF-SVM, an exponentially growing sequence of C and γ are tried from 2^{-4} to 2^{8}.

4.2. Experimental Results

Table 2 summarizes a comparison of character recognition rates (in %) on test set of LBG-trained MP-based classifiers with 1 or 2 prototypes, SVM-based classifiers with linear or RBF kernel functions, and MSM-trained MP-based classifiers with 1 or 2 prototypes on three two-class classification tasks (A, B, C) and one ten-class classification task (*All*, including A, B, C, D, E). It is observed that 1) MSM-trained classifiers with single prototype achieve a similar performance with that of linear-SVM-based classifiers, 2) MSM-trained classifiers with 2 prototypes perform better than that of RBF-kernel-based SVM classifiers on two-class classification tasks, 3) MSM-trained classifier with 2 prototypes achieves a similar performance with that of RBF-kernel-based SVM classifier on the ten-class classification task. An important advantage of MP-based classifier is that the number of parameters is far smaller than that of SVM-based classifier, where there are more than 60 support vectors for each class in RBF-kernel-based SVMs.

5. SUMMARY

In this paper, we have proposed a maximum separation margin (MSM) method for training multiple-prototype-based pattern classifiers, and studied how to use SDP techniques to solve the MSM training problem. The effectiveness of the proposed approach has been confirmed by evaluation results on several tasks of recognition of confusable handwritten Kanji characters. Ongoing and future works include 1) to reformulate the training problem appropriately that can reflect the original intention of using the notion of separation margin, yet the problem can be solved by using some more scalable optimization techniques rather than SDP; and 2) to evaluate the above techniques on large vocabulary handwritten Chinese character recognition tasks.

6. REFERENCES

[1] Y. Altun, I. Tsochantaridis, and T. Hofmann, "Hidden Markov support vector machines," *Proc. ICML-2003*, Washington DC., 2003.

[2] Z.-L. Bai and Q. Huo, "A study on the use of 8-directional features for online handwritten Chinese character recognition," *Proc. ICDAR-2005*, pp.262-266.

[3] S. J. Benson and Y. Ye, *DSDP5: Software For Semidefinite Programming*, Mathematics and Computer Science Division, Argonne National Laboratory, Preprint ANL/MCS-P1289-0905, 2005, (available at http://www-unix.mcs.anl.gov/DSDP/).

[4] C.-C. Chang and C.-J. Lin, *LIBSVM: A Library for Support Vector Machines*, 2001, (available at http://www.csie.ntu.edu.tw/ cjlin/libsvm/).

[5] H. Jiang, X.-W. Li and C.-J. Liu, "Large margin hidden Markov models for speech recognition," *IEEE Trans. Audio, Speech, and Language Processing*, vol.14, no.5, pp.1584-1595, 2006.

[6] J.-Y. Li, M. Yuan, and C.-H. Lee, "Soft margin estimation of hidden Markov model parameters," *Proc. Interspeech 2006 – ICSLP*, pp.2422-2425.

[7] X.-W. Li and H. Jiang, "Solving large margin estimation of HMMs via semidefinite programming," *Proc. Interspeech 2006 – ICSLP*, pp.2414-2417.

[8] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. on Communication*, vol. COM-28, pp.84-95, 1980.

[9] M. Nakagawa and K. Matsumoto, "Collection of on-line handwritten Japanese character pattern databases and their analysis," *International Journal on Document Analysis and Recognition*, vol. 7, pp.69-81, 2004.

[10] F. Sha and L. Saul, "Large margin Gaussian mixture modeling for phonetic classification and recognition," *Proc. ICASSP-2006*, pp.I-265-268.

[11] V. Vapnik, *Statistical Learning Theory*, Wiley, 1998.

[12] Y. Ye, "Semidefinite programming for sensor network and graph localization," in *Robust Optimization-Directed Design*, A. J. Kurdila and P. M. Pardalos (Eds.), Springer, 2006, pp.247-275.

[13] D. Yu, L. Deng, X.-D. He, and A. Acero, "Use of incrementally regulated discriminative margins in MCE training for speech recognition," *Proc. Interspeech 2006 – ICSLP*, pp.2418-2421.