Support Cluster Machine

Bin Li LIBIN@FUDAN.EDU.CN
Mingmin Chi MMCHI@FUDAN.EDU.CN
Department of Computer Science and Engineering, Fudan University, Shanghai, 200433 P. R. China

Jianping Fan JFAN@UNCC.EDU
Department of Computer Science, UNC-Charlotte, Charlotte, NC 28223 USA

Xiangyang Xue XYXUE@FUDAN.EDU.CN
Department of Computer Science and Engineering, Fudan University, Shanghai, 200433 P. R. China

Abstract

For large-scale classification problems, the training samples can be clustered beforehand as a downsampling pre-process, and then only the obtained clusters are used for training. Motivated by such assumption, we proposed a classification algorithm, Support Cluster Machine (SCM), within the learning framework introduced by Vapnik. For the SCM, a compatible kernel is adopted such that a similarity measure can be handled not only between clusters in the training phase but also between a cluster and a vector in the testing phase. We also proved that the SCM is a general extension of the SVM with the RBF kernel. The experimental results confirm that the SCM is very effective for large-scale classification problems due to significantly reduced computational costs for both training and testing and comparable classification accuracies. As a by-product, it provides a promising approach to dealing with privacy-preserving data mining problems.

1. Introduction

Researchers have made great efforts to improve the efficiency of SVM for large-scale classification problems via various approaches, e.g., decomposition methods (Osuna et al., 1997; Joachims, 1999; Platt, 1999; Collobert & Bengio, 2001; Keerthi et al., 2001), incremental algorithms (Cauwenberghs & Poggio, 2000; Fung & Mangasarian, 2002; Laskov et al., 2006), parallel techniques (Collobert et al., 2001; Graf et al., 2004), and employing an approximate formula (Fung & Mangasarian, 2001; Lee & Mangasarian, 2001).

Another approach, in which only the representatives are used for training, also aims at large-scale classification problems. Active learning (Schohn & Cohn, 2000) chooses the representatives simply by the heuristic. CB-SVM (Yu et al., 2003) recursively selects the centroids of the clusters as the representatives along the hierarchical clustering tree. In (Sun et al., 2004), it is assumed that the samples residing on the boundaries of the clusters are critical data, thus only these samples are used for training. Core Vector Machine (Tsang et al., 2005) chooses the core set by solving a minimum enclosing ball problem. In (Boley & Cao, 2004; Yuan et al., 2006), the potential support vectors (SVs) are first identified by an approximate classifier trained on the centroids of the clusters, and then, the clusters are replaced by the SVs (Boley & Cao, 2004), or the non-SVs are removed (Yuan et al., 2006).

In this paper, we propose the Support Cluster Machine (SCM) to effectively deal with large-scale classification problems. The proposed algorithm follows the learning framework introduced by (Vapnik, 1998), and addresses the difficulties in the kernel space. The main idea of the SCM is as follows. The training samples (feature vectors) are clustered beforehand as a downsampling pre-process, and the obtained clusters are taken as the training units, whereas the testing samples are still the original feature vectors. Then, a compatible kernel is employed for the SCM such that a similarity measure can be handled not only between clusters in the training phase but also between a cluster and a vector in the testing phase. Thus, the sup-

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007.

port clusters obtained in the training phase can be directly used in the decision function. The experiments carried out on the toy dataset and large-scale datasets confirm the effectiveness of the proposed algorithm by providing the comparable classification accuracies as well as by significantly reducing the computational costs for both training and testing. Besides, the SCM can be interestingly applied to the privacy-preserving data mining tasks.

Compared to the existing related algorithms, the SCM has the following advantages: (a) it is able to achieve a comparable classification accuracy, but with significantly reduced computational and spatial costs for both training and testing; (b) it compresses the training samples by adopting the generative models as the training units, which are able to keep the complete statistical information of the original data, rather than selecting a set of representatives; and (c) it is easily implemented.

The remainder of the paper is organized as follows. In Section 2, the learning framework of the proposed SCM and the probability product kernel is firstly described, and then the proposed SCM is presented in detail including the training and the testing phases. Section 3 reports the experimental results for large-scale classification problems and privacy-preserving problems. Finally, we make the conclusion and the discussion on the work in Section 4.

2. The Proposed Algorithm

The training samples are given as $N = N^+ + N^-$ pairs $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, where the feature vector $\mathbf{x}_i \in \mathbb{R}^D$ and the label $y_i \in \{1, -1\}$.

For the SCM training, N^+ positive and N^- negative samples are clustered, respectively. Then, the $K = K^+ + K^-$ training clusters can be obtained. We assume that these clusters follow Gaussian distributions, then the training clusters can be denoted as a set of training pairs $\mathcal{C} = \{(\Theta_k, y_k)\}_{k=1}^K$, where the generative model $\Theta_k = (P_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ comprises the prior (weight), the mean, and the covariance matrix, of the kth cluster. In the case of training units being clusters, all the training clusters should satisfy the following constraints:

$$y_k\big(\mathbf{w}^\top \phi(\Theta_k) + b\big) \geq 1 - \xi_k, \quad k = 1, \ldots, K \tag{1}$$

where $\phi(\cdot)$ is a mapping function that projects the generative model into the infinite dimensional space of probability distributions.

Like the SVM, the SCM also follows the principle of Structural Risk Minimization (SRM) by introducing a regularization term into the cost function to minimize the VC bound, i.e., maximize the margin between the positive and the negative clusters:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\mathbf{w}^\top \mathbf{w} + C\sum_{k=1}^K P_k \xi_k \tag{2}$$

where the slack ξ_k is multiplied by the weight P_k (the prior of the kth cluster) such that a misclassified cluster with more samples could be given heavier penalty.

Incorporating the constraints (1) and the constraints $\xi_k \geq 0, k = 1, \ldots, K$, to the cost function (2), the constrained optimization problem is formulated into the primal Lagrangian. Then, the primal is transformed to the dual problem following the same steps as in the SVM. For the space limitation, we directly give the dual representation of the SCM as follows:

$$\max_{\boldsymbol{\alpha}} \sum_{k=1}^K \alpha_k - \frac{1}{2}\sum_{k=1}^K\sum_{l=1}^K y_k y_l \alpha_k \alpha_l \phi(\Theta_k)^\top \phi(\Theta_l) \tag{3}$$

$$\text{s.t.} \quad 0 \leq \alpha_k \leq P_k C, \quad k = 1, \ldots, K$$
$$\sum_{k=1}^K \alpha_k y_k = 0.$$

In the testing phase, we treat the testing sample $\mathbf{x}$ as an extreme case of the cluster $\Theta_\mathbf{x}$ when its covariance matrix vanishes, i.e., $\Theta_\mathbf{x} = (P_\mathbf{x} = 1, \boldsymbol{\mu}_\mathbf{x} = \mathbf{x}, \boldsymbol{\Sigma}_\mathbf{x} = \mathbf{0})$. After solving the above dual problem, the coefficients $\alpha_k, k = 1, \ldots, K$, can be obtained. Then, a testing sample $\mathbf{x}$ can be predicted by using the following decision function:

$$f(\mathbf{x}) = \operatorname{sgn}\big(\sum_{k=1}^K \alpha_k y_k \phi(\Theta_k)^\top \phi(\Theta_\mathbf{x}) + b\big). \tag{4}$$

It is worth noting that the SCM has the same framework as in the SVM. However, the dual optimization problem (3) and the decision function (4) of the SCM have shown that, the remaining difficulties lie in finding a compatible kernel which is able to measure the similarity not only between two clusters in the training phase but also between a cluster and a sample vector in the testing phase. In the following subsections, we will give the details for addressing these difficulties.

2.1. Probability Product Kernel

One valid kernel which satisfies the requirements of the SCM is derived from the probability product kernel. Here, we only give a brief introduction to the probability product kernel with the Gaussian distribution. For details, one can refer to (Jebara et al., 2004).

The probability product kernel between two distributions p_k and p_l is defined as:

$$\kappa_\rho(p_k, p_l) = \int_{\mathbb{R}^D} p_k^\rho p_l^\rho d\mathbf{x} \tag{5}$$

where $\kappa_\rho(p_k, p_l)$ is positive definite, and the exponential ρ leads to a set of candidate kernels. We choose $\rho = 1$ in order to facilitate the application of the underlying kernel to the testing phase. The interesting result induced by setting $\rho = 1$ is given in Section 2.3.

When p_k and p_l are both Gaussian distributions, i.e., $p_k = P_k p(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ and $p_l = P_l p(\mathbf{x}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$, $\kappa(p_k, p_l)$ can be written as $\kappa(\Theta_k, \Theta_l)$, which is a function of two generative models. Then, $\kappa(\Theta_k, \Theta_l)$ can be computed directly using the corresponding parameters of two models to avoid integrating the probability distributions in the entire input space. Hence, we have:

$$\begin{aligned} & \kappa(\Theta_k, \Theta_l) \qquad (6) \\ = \; & P_k P_l \int_{\mathbb{R}^D} p(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) p(\mathbf{x}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) d\mathbf{x} \\ = \; & P_k P_l (2\pi)^{-\frac{D}{2}} |(\boldsymbol{\Sigma}_k^{-1} + \boldsymbol{\Sigma}_l^{-1})^{-1}|^{\frac{1}{2}} |\boldsymbol{\Sigma}_k|^{-\frac{1}{2}} |\boldsymbol{\Sigma}_l|^{-\frac{1}{2}} \\ & \exp\big(-\frac{1}{2}(\boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}_k^{-1} \boldsymbol{\mu}_k + \boldsymbol{\mu}_l^\top \boldsymbol{\Sigma}_l^{-1} \boldsymbol{\mu}_l - \widetilde{\boldsymbol{\mu}}^\top \widetilde{\boldsymbol{\Sigma}}^{-1} \widetilde{\boldsymbol{\mu}})\big) \end{aligned}$$

where $\widetilde{\boldsymbol{\Sigma}}^{-1} = (\boldsymbol{\Sigma}_k^{-1} + \boldsymbol{\Sigma}_l^{-1})^{-1}$, $\widetilde{\boldsymbol{\mu}} = \boldsymbol{\Sigma}_k^{-1}\boldsymbol{\mu}_k + \boldsymbol{\Sigma}_l^{-1}\boldsymbol{\mu}_l$.

2.2. Training Phase

To learn support clusters for the SCM, a clustering technique (e.g., K-Means (Hartigan & Wong, 1979), the Expectation Maximization (EM) algorithm (Dempster et al., 1977), hierarchical clustering (Zhang et al., 1996)) should be applied on the training samples beforehand. This is the first step of the training phase. After clustering, (3) is exploited for the SCM learning. This consists of the second step of the training phase. In what follows, the two steps will be discussed in detail.

2.2.1. Clustering

In this step, we can select a proper clustering technique for the task observed. For the SCM, clustering is employed to compress the training samples, where a generative model is used to represent a local subset of the training samples. Accordingly, the clusters are not necessary to fit the density of the data very well. For large-scale classification problems, it is also crucial to reduce the computational complexity. Therefore, any efficient clustering technique, which is able to approximately describe the data layout in the input space, can be adopted for the SCM.

In our work, we adopt a clustering technique named Threshold Order-Dependent (TOD) algorithm (Friedman & Kandel, 1999). The main idea of the TOD algorithm is as follows. The training samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots\}$ are fed sequentially. For the given sample $\mathbf{x}_i$, if the longest distance from $\mathbf{x}_i$ to the centers of the existing clusters exceeds a predefined threshold, a new cluster with the center $\mathbf{x}_i$ is created; otherwise, $\mathbf{x}_i$ belongs to the cluster whose center is closest to it. The TOD algorithm has two attractive properties: 1) a linear computational complexity, and 2) being able to deal with *sequential* data with a negligible spatial complexity. We also used the EM algorithm for comparison. On the analysis of the experimental results in Section 3.3, one can see that the TOD algorithm is more appropriate for the SCM.

2.2.2. SCM Learning

After clustering, the training clusters $\{(\Theta_k, y_k)\}_{k=1}^K$ are obtained. Like in the SVM, the inner product in (3) can be replaced by the kernel (6). Hence, we can get the following objective function:

$$\max_{\boldsymbol{\alpha}} \sum_{k=1}^K \alpha_k - \frac{1}{2} \sum_{k=1}^K \sum_{l=1}^K y_k y_l \alpha_k \alpha_l \kappa(\Theta_k, \Theta_l). \tag{7}$$

In practice, we can simply use the diagonal entries of the covariance matrices, i.e., $\boldsymbol{\Sigma}_k = \mathrm{diag}(\big(\sigma_k^{(1)}\big)^2, \ldots, \big(\sigma_k^{(D)}\big)^2)$, to avoid computing the inverse matrices in (6). Thus, the kernel becomes:

$$\begin{aligned} & \kappa(\Theta_k, \Theta_l) \\ = \; & P_k P_l \exp\left\{ -\frac{1}{2} \sum_{d=1}^D \frac{\big(\mu_k^{(d)} - \mu_l^{(d)}\big)^2}{\big(\sigma_k^{(d)}\big)^2 + \big(\sigma_l^{(d)}\big)^2} \right\} \Big/ \\ & \prod_{d=1}^D \sqrt{2\pi\Big(\big(\sigma_k^{(d)}\big)^2 + \big(\sigma_l^{(d)}\big)^2\Big)}. \end{aligned} \tag{8}$$

The priors of the training clusters are defined in the following two ways corresponding to different clustering techniques: 1) *hard ownership* (e.g., the TOD algorithm): $P_k = \frac{N_k}{N}$; 2) *soft (probability) ownership* (e.g., the EM algorithm): $P_k = \frac{P_k^+ N^+}{N}$ ($P_k = \frac{P_k^- N^-}{N}$), where P_k^+ (P_k^-) denotes the normalized weight in the original positive (negative) GMM obtained in the first step.

2.3. Testing Phase

The proposed algorithm is able to measure the similarity between a cluster and a vector by employing a compatible kernel. Here, the testing sample $\mathbf{x}$ can be viewed as the extreme case of the Gaussian distribution, where the prior is fixed and the covariance matrix

vanishes, i.e., $\Theta_{\mathbf{x}} = (P_{\mathbf{x}} = 1, \boldsymbol{\mu}_{\mathbf{x}} = \mathbf{x}, \boldsymbol{\Sigma}_{\mathbf{x}} = \mathbf{0})$. Thus, the support clusters obtained in the SCM learning step can be directly used in the decision function.

Theorem 1. *Given two Gaussians, Θ_k and Θ_l, if the prior $P_l = 1$, and the covariance matrix $\boldsymbol{\Sigma}_l$ vanishes, i.e., $\boldsymbol{\Sigma}_l \to \mathbf{0}$, then, the limit of the kernel function* (6) *becomes the posterior probability of $\boldsymbol{\mu}_l$ given Θ_k.*

Proof. By setting the exponential $\rho = 1$ (see Section 2.1), the kernel (6) can be written in the form of the expectation of p_k under p_l, i.e., $\mathrm{E}_{p_l}[p_k]$. Then, the limit of the expectation as $\boldsymbol{\Sigma}_l$ vanishes gives:

$$\begin{aligned} & \lim_{\boldsymbol{\Sigma}_l \to \mathbf{0}} \mathrm{E}_{p_l}[p_k] \\ = \; & P_k P_l p(\boldsymbol{\mu}_l|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \lim_{\boldsymbol{\Sigma}_l \to \mathbf{0}} \int_{\boldsymbol{\mu}_l^\epsilon} p(\mathbf{x}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) d\mathbf{x} \\ = \; & P_k p(\boldsymbol{\mu}_l|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \end{aligned} \tag{9}$$

where $\boldsymbol{\mu}_l^\epsilon$ is the ϵ-neighborhood of $\boldsymbol{\mu}_l$, i.e., the open D-ball with radius ϵ centered at $\boldsymbol{\mu}_l$, $\epsilon > 0$. $\forall \mathbf{x} \in \boldsymbol{\mu}_l^\epsilon$, the probability density, $p(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, can be viewed as the constant value $p(\boldsymbol{\mu}_l|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$. As $\boldsymbol{\Sigma}_l \to \mathbf{0}$, the probability integral of $p(\mathbf{x}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$ over $\mathbf{x}$ in $\boldsymbol{\mu}_l^\epsilon$ equals to 1, and the result can be derived. □

We replace Θ_l in (9) by the testing sample $\Theta_{\mathbf{x}}$, and get the kernel for the SCM testing:

$$\kappa(\Theta_k, \Theta_{\mathbf{x}}) = P_k p(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{10}$$

which is the posterior probability of $\mathbf{x}$ given Θ_k. By substituting (10) for the inner product in (4), then the decision function of the SCM gives:

$$f(\mathbf{x}) = \mathrm{sgn}\big(\sum_{k=1}^{K} \alpha_k y_k P_k p(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) + b\big). \tag{11}$$

For simplicity, like in the training phase we can also use the diagonal entries of the covariance matrices in the testing phase. We replace Θ_l in (8) by $\Theta_{\mathbf{x}}$, and set $\sigma_{\mathbf{x}}^{(d)} = 0, d = 1, \ldots, D$. Then, we also get the posterior probability of $\mathbf{x}$ given Θ_k:

$$\begin{aligned} & \kappa(\Theta_k, \Theta_{\mathbf{x}}) \\ = \; & P_k \prod_{d=1}^{D} \frac{1}{\sqrt{2\pi}\sigma_k^{(d)}} \exp\left\{ -\frac{1}{2} \sum_{d=1}^{D} \frac{\big(\mu_k^{(d)} - \mathbf{x}^{(d)}\big)^2}{\big(\sigma_k^{(d)}\big)^2} \right\}. \end{aligned} \tag{12}$$

2.4. Connection with SVMs

Since the proposed SCM and the SVM with the RBF kernel both handle the basic training units with Gaussian distributions, there should exist an underlying connection between these two algorithms.

Lemma 1. *If the covariance matrices of two clusters take $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}_l = \sigma^2 \mathbf{I}$, then* (6) *becomes the Radius Basis Function (RBF) kernel (Jebara et al., 2004):*

$$\kappa(\Theta_k, \Theta_l) = \frac{P_k P_l}{(4\pi\sigma^2)^{\frac{D}{2}}} \exp\big(-\frac{\|\boldsymbol{\mu}_k - \boldsymbol{\mu}_l\|^2}{4\sigma^2}\big). \tag{13}$$

Theorem 2. *The SVM with the RBF kernel is the extreme case of the SCM when each training cluster in the SCM comprises only one sample.*

Proof. In the case that each training cluster comprises only one sample, both the training clusters and the testing sample should be viewed *equally* as such Gaussians, where $P_k = P_{\mathbf{x}} = 1$, and $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}_{\mathbf{x}} = \sigma^2 \mathbf{I}, k = 1, \ldots, K$. Then, the kernel both for the SCM training and testing becomes the RBF kernel:

$$\kappa(\Theta_k, \Theta_l) = \lambda \exp(-\gamma \|\boldsymbol{\mu}_k - \boldsymbol{\mu}_l\|^2) \tag{14}$$

where $\lambda = (4\pi\sigma^2)^{-\frac{D}{2}}$ and $\gamma = \frac{1}{4\sigma^2}$ are constants. In addition, the SCM follows the same learning framework as the SVM does. Thus, the SCM in the extreme case reduces to the SVM with the RBF kernel. □

As aforementioned, the SCM can be viewed as a general extension of the SVM with the RBF kernel by introducing the size (prior) and the shape (covariance) information into a training unit.

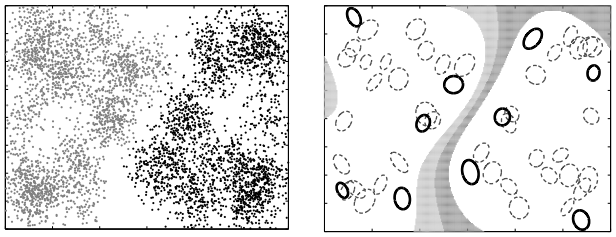
3. Experimental Results

We have performed sets of experiments on a toy dataset and two real databases to demonstrate the effectiveness of the SCM: (a) **Toydata**. This synthetic dataset was randomly generated for visualizing the learning result of the SCM in the 2-D space. (b) **MNIST**[1]. This is a large-scale benchmark database with 10 classes of handwritten digits for classification tasks. (c) **Adult**[2]. This is also a large-scale benchmark database and the task is to predict whether income exceeds \$50K/yr based on census data. Based on this database, we simulated a privacy-preserving task to show an interesting application of the SCM.

For comparison, we adopted three state-of-the-art implementations for SVM, i.e., libsvm (SMO-type algorithm) (Chang & Lin, 2001), SVMTorch (Collobert & Bengio, 2001), SVM^{light} (Joachims, 1999), and CVM (Tsang et al., 2005). All the experiments were performed on a $3.0GHz$ CPU.

[1] Available at http://yann.lecun.com/exdb/mnist/.

[2] Available at http://www.ics.uci.edu/mlearn/MLRepository.html.



(a) (b)

Figure 1. (a) Samples. (b) Learning result of the SCM on the 2-D toydata; the ellipses in black bold denote the support clusters, and the gray area denotes the margin.

3.1. Model Selection

For the SCM, the training clusters have already had covariances, it is not required to select γ in the training phase, thus there remains only one parameter C for the model selection by grid search. For the four comparing methods, there are two parameters, C and γ, we selected the optimal pair by grid search. In our experiments, we adjusted C in the range of $\{10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\}$, and γ in the range of $\{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$.

3.2. Toydata

We generated a two-class 2-D synthetic dataset for visualizing the learning result of the SCM. For each class, 2500 samples were generated by following a mixture of Gaussian distributions which was predefined (cf. Figure 1(a)). We first employed the TOD algorithm to cluster the positive and the negative samples respectively, and obtained 25 positive and 25 negative clusters. Then, we calculated the means, the weights (priors), and the covariance matrices for the clusters. In Figure 1(b), the clusters are denoted by ellipses, whose sizes are proportional to the weights.

We used the proposed algorithm to learn a classifier on the training clusters. In Figure 1(b), the support clusters are labeled in black bold, which determine the margin of the SCM (light gray area $-1 \leq f(\mathbf{x}) < 0$, deep gray area $0 \leq f(\mathbf{x}) \leq 1$). One can see that the SCM also tries to maximize the margin between two classes like that the SVM does.

3.3. Large-scale Dataset: MNIST

MNIST is a large-scale benchmark database which is usually used for evaluating classification performance. The dataset comprises 10 classes, handwritten digits '0'–'9', in total 60000 samples for training (about 6000 for each class) and 10000 samples for testing. In our experiments, we investigated the characteristics of the SCM by analyzing the average results of 45 binary classifiers. We also compared the results obtained by the SCM with those by the four comparing methods.

Table 1. The average performance of the SCM on the MNIST database compared to three SVMs and CVM. Note that the training time by the SCM comprises both the clustering time and the SCM learning time.

Methods	Training (s)	Testing (s)	Error (%)
libsvm	5.78	1.77	0.31
SVMTorch	3.96	7.20	0.31
SVM^{light}	19.5	4.46	0.31
CVM	27.9	1.21	0.31
SCM+TOD	**0.78**	**0.27**	**1.07**
SCM+EM	9.50	0.08	1.47

Table 1 reports the training times, the testing times, and the average testing errors obtained by the four comparing methods as well as the SCMs with different clustering algorithms (i.e., the TOD algorithm and the EM algorithm). Since the number of the training clusters and the support clusters in the SCMs are both fewer than that of the training samples and the SVs in the SVMs, the learning and the testing time for the SCMs are significantly less than the one for the SVMs. For the SCM, the clustering technique has significant influence on training times and testing errors. By analyzing Table 1, one can see that the SCM with the TOD algorithm takes the least training time (i.e., 0.78s) and testing time (i.e., 0.27s), and obtains a comparable testing error 1.07%. In contrast, the SCM with the EM algorithm does not work as well as the one with the TOD algorithm. Due to the higher computational cost of the EM algorithm, we set a smaller K and obtained only about 50 training clusters, which are insufficient to describe the data layout in the input space. Therefore, the TOD clustering algorithm is more effective for the SCM.

Let us look closer on the results obtained by the SCM with the TOD algorithm. Figure 2(a) plots the average testing error with respect to the number of training clusters K. In our experiments, K was adjusted by tuning the threshold of the TOD algorithm. With K increasing, the average testing error reduces gradually, whereas the average time for the SCM learning increases polynomially (cf. Figure 2(b)) and the average number of the support clusters increases sublinearly (cf. Figure 2(c)). Ideally, these curves will keep their trends until arrive at $K = N$, i.e., the extreme case that each cluster comprises only one sample. However,

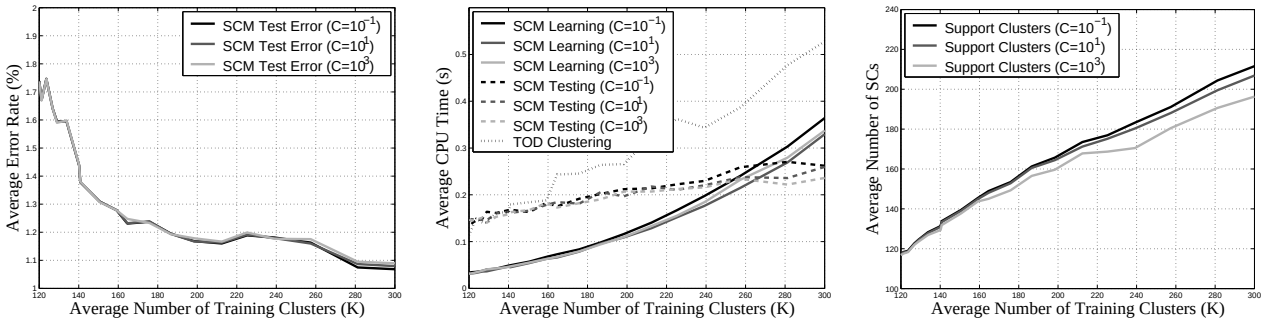


Figure 2. Average testing results of the SCM on the MNIST database: (a) average testing error rate; (b) average CPU time for training and testing; (c) average number of support clusters (SCs).

at the point around 280 clusters, the SCM is able to achieve the testing errors about 1.0%, which are comparable to the results obtained by the SVMs. In the meanwhile, by analyzing the curves in Figure 2, one can observe that the regularization parameter C has no significant influence on the classification results. This confirms the robustness of the proposed algorithm.

It is worth noting that in the high-dimensional space, the number of the training clusters is always equal to or even less than the number of dimensions of the input space. In this case, a higher ratio of the training clusters can not be surrounded by the other training clusters of the same class, i.e., are located on the boundary, hence they are more likely to work as the support clusters; whereas in the low-dimensional space, the majority of the training clusters can be surrounded by few support clusters, e.g., the toydata in Figure 1(b). The above analysis explains why there exists relatively higher ratio of the number of support clusters to the number of training clusters in Figure 2(c).

3.4. Privacy-preserving Dataset: Adult

Adult database contains 30162 training samples and 15060 test samples (the samples with unknown values were removed), and the percentage of the positive samples is 24.78%. We used the Adult database to simulate a common task in privacy-preserving data mining on the *horizontally*[3] partitioned data, i.e., to learn a prediction rule from a set of databases without disclosing the individual samples from one party to the others (and the public). We partitioned the training samples into three subsets in an equal size (cf. Table 2), which are viewed as the databases of three parties. Since each party would not like to disclose the individual information, one cannot simply combine the three databases for training.

The proposed SCM can be used to deal with such observed privacy-preserving tasks. With the aid of the SCM, each party is only required to offer a set of training clusters (represented by generative models) for training. In the simulated task, we first used the TOD algorithm to cluster the training samples from three datasets respectively, and obtained three positive and three negative GMMs. Then, we combined the positive and the negative GMMs into one positive and one negative GMM, by resetting the priors of the training clusters as follows:

$$\hat{P}_k^{(m)} = \frac{P_k^{(m)} N^{(m)}}{\sum_n N^{(n)}} \tag{15}$$

where $P_k^{(m)}$ denotes the prior of the kth cluster in the mth database, and $N^{(m)}$ denotes the number of training samples in the mth database. Finally, we used the proposed algorithm to learn the classifiers on the combined training clusters.

Besides the result obtained by the SCM on the partitioned datasets, we also provides the results obtained by the SCM and the four comparing methods on the complete (non-partitioned) Adult database, which are reported in Table 3. One can see that the SCM is able to preserve the data privacy, in the meanwhile, achieves comparable classification accuracy provided by the state-of-the-art implementations for SVM.

4. Conclusion and Discussion

In this paper, we have proposed the Support Cluster Machine (SCM) to address large-scale classification problems. Moreover, it also can be applied to privacy-preserving data mining problems. The SCM deals with these two kinds of problems by compressing the train-

[3]Each party has a subset of the data with full attributes.

Table 2. The Adult database is partitioned into three subsets to simulate three databases from three parties.

Party	Positive	Negative
1	2463	7587
2	2493	7557
3	2552	7510

Table 3. The classification results on the Adult database. Note that (p) indicates the SCM is applied to the simulated privacy-preserving task.

Methods	Database partitioned?	Error (%)
libsvm	×	16.27
SVMTorch	×	16.51
SVM^{light}	×	15.70
CVM	×	15.98
SCM	×	16.66
SCM (p)	√	**16.83**

ing samples in order to 1) downsample the data for the large-scale problems, and 2) hide the individual information for the privacy-preserving problems. The proposed algorithm compresses the training samples via a new approach, i.e., taking generative models as the training units; consequently, the SCM can be viewed as a general extension of the SVM with the RBF kernel by introducing the size (prior) and the shape (covariance) information into a training unit. The remaining difficulty, i.e., measuring the similarity between a cluster and a vector, has also been addressed by adopting a compatible kernel which is derived from the probability product kernel; thus, the support clusters can be directly used in the prediction function.

The SCM is different from the existing related methods in the following aspects: (a) For the SCM, the training units are generative models while the testing units are vectors; however, for the other methods, both the training and the testing units are vectors or "supervectors" (e.g., GMMs (Moreno et al., 2003; Campbell et al., 2006)). (b) In the SCM, the training units contain the complete statistical information of the original data; whereas in the "selective sampling" methods, the selected representatives may lose the statistical information. (c) In the SCM, there is only one parameter C for model selection; but in the other methods, there are usually two or even more parameters. (d) The proposed algorithm can be easily implemented; while most of the existing algorithms are more complicated.

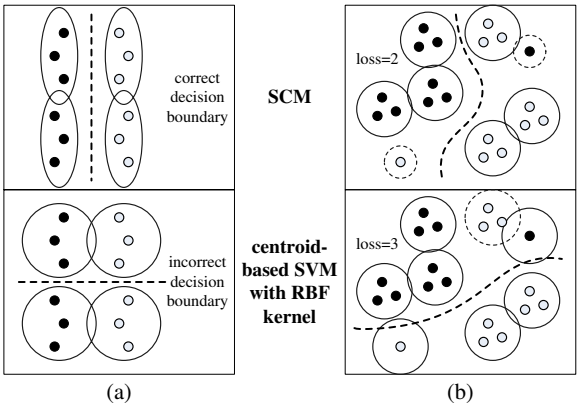


Figure 3. The advantages of using (a) covariances and (b) priors in the SCM. In (b), although the SCM has more misclassified clusters than the centroid-based SVM, its loss is smaller since the misclassified clusters in the SCM usually comprise fewer data.

By employing the covariances (second-order information) and priors of clusters, the SCM can improve the accuracy compared to the SVM which only uses the centroids of clusters as the representatives. The advantage of using covariances is that they can lead to a more accurate similarity measure between clusters (cf. Figure 3(a)). The advantage of using priors is that they can force the learner to concern more the clusters which comprise more data (cf. Figure 3(b)). However, since the SCM uses covariances in place of widths in RBF kernels, the VC dimension can not be adjusted like using RBF kernel (larger width leads to lower VC dimension). Therfore, we will further investigate the generalization ability of the SCM in the future work.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable suggestions. This work was supported in part by Natural Science Foundation of China under contracts 60533100 and 60402007, and Shanghai Municipal R&D Foundation under contracts 05QMH1403, 065115017 and 06DZ15008.

References

Boley, D., & Cao, D. (2004). Training support vector machine using adaptive clustering. *Proc. of the SIAM Int'l Conf. on Data Mining.*

Campbell, W. M., Sturim, D. E., & Reynolds, D. A. (2006). Support vector machines using GMM supervectors for speaker verification. *IEEE Signal Pro-*

cessing Letters, *13*, 308–311.

Cauwenberghs, G., & Poggio, T. (2000). Incremental and decremental support vector machine learning. *Advanced Neural Information Processing Systems*. Cambridge, MA: MIT Press.

Chang, C., & Lin, C. (2001). *LIBSVM: A library for support vector machines*.

Collobert, R., & Bengio, S. (2001). SVMTorch: Support vector machines for large-scale regression problems. *J. of Machine Learning Research*, *1*, 143–160.

Collobert, R., Bengio, S., & Bengio, Y. (2001). A parallel mixture of SVMs for very large scale problems. *Advanced Neural Information Processing Systems*. Cambridge, MA: MIT Press.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, *39*, 1–38.

Friedman, M., & Kandel, A. (1999). *Introduction to pattern recognition*, chapter Distance Functions, 70–73. London, UK: Imperial College Press.

Fung, G., & Mangasarian, O. L. (2001). Proximal support vector machine classifiers. *Proc. of the ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining* (pp. 77–86).

Fung, G., & Mangasarian, O. L. (2002). Incremental support vector machine classification. *Proc. of the SIAM Int'l Conf. on Data Mining*.

Graf, H. P., Cosatto, E., Bottou, L., Durdanovic, I., & Vapnik, V. (2004). Parallel support vector machines: The cascade SVM. *Advanced Neural Information Processing Systems*. Cambridge, MA: MIT Press.

Hartigan, J. A., & Wong, M. A. (1979). A K-Means clustering algorithm. *Applied Statistics*, *28*, 100–108.

Jebara, T., Kondor, R., & Howard, A. (2004). Probability product kernels. *J. of Machine Learning Research*, *5*, 819–844.

Joachims, T. (1999). Making large-scale svm learning practical. In B. Schölkopf, C. J. C. Burges and A. J. Smola (Eds.), *Advances in kernel methods - support vector learning*, 169–184. Cambridge, MA: MIT Press.

Keerthi, S. S., Shevade, S. K., Bhattacharyya, C., & Murthy, K. R. K. (2001). Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation*, *13*, 637–649.

Laskov, P., Gehl, C., Krüger, S., & Müller, K. (2006). Incremental support vector learning: Analysis, implementation and applications. *J. of Machine Learning Research*, *7*, 1909–1936.

Lee, Y., & Mangasarian, O. L. (2001). RSVM: Reduced support vector machines. *Proc. of the SIAM Int'l Conf. on Data Mining*.

Moreno, P. J., Ho, P. P., & Vasconcelos, N. (2003). A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. *Advanced Neural Information Processing Systems*. Cambridge, MA: MIT Press.

Osuna, E., Freund, R., & Girosi, F. (1997). An improved training algorithm for support vector machines. *Proc. of the 1997 IEEE Workshop on Neural Networks for Signal Processing* (pp. 276–285).

Platt, J. (1999). Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges and A. J. Smola (Eds.), *Advances in kernel methods - support vector learning*, 185–208. Cambridge, MA: MIT Press.

Schohn, G., & Cohn, D. (2000). Less is More: Active learning with support vector machines. *Proc. of the Int'l Conf. on Machine Learning*.

Sun, S., Tseng, C. L., Chen, Y. H., Chuang, S. C., & Fu, H. C. (2004). Cluster-based support vector machines in text-independent speaker identification. *Proc. of the Int'l Joint Conf. on Neural Network*.

Tsang, I. W., Kwok, J. T., & Cheung, P. (2005). Core Vector Machines: Fast SVM training on very large data sets. *J. of Machine Learning Research*, *6*, 363–392.

Vapnik, V. (1998). *Statistical learning theory*. John Wiley.

Yu, H., Yang, J., & Han, J. (2003). Classifying large data sets using SVMs with hierarchical clusters. *Proc. of the ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining* (pp. 306–315).

Yuan, J., Li, J., & Zhang, B. (2006). Learning concepts from large scale imbalanced data sets using support cluster machines. *Proc. of the ACM Int'l Conf. on Multimedia* (pp. 441–450).

Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: An efficient data clustering method for very large databases. *Proc. of the 1996 ACM SIGMOD Int'l Conf. on Management of Data* (pp. 103–114).