Clustered Support Vector Machines

Quanquan Gu
Department of Computer Science
University of Illinois at Urbana-Champaign
qgu3@illinois.edu

Jiawei Han
Department of Computer Science
University of Illinois at Urbana-Champaign
hanj@cs.uiuc.edu

Abstract

In many problems of machine learning, the data are distributed nonlinearly. One way to address this kind of data is training a nonlinear classifier such as kernel support vector machine (kernel SVM). However, the computational burden of kernel SVM limits its application to large scale datasets. In this paper, we propose a Clustered Support Vector Machine (CSVM), which tackles the data in a divide and conquer manner. More specifically, CSVM groups the data into several clusters, followed which it trains a linear support vector machine in each cluster to separate the data locally. Meanwhile, CSVM has an additional global regularization, which requires the weight vector of each local linear SVM aligning with a global weight vector. The global regularization leverages the information from one cluster to another, and avoids over-fitting in each cluster. We derive a data-dependent generalization error bound for CSVM, which explains the advantage of CSVM over linear SVM. Experiments on several benchmark datasets show that the proposed method outperforms linear SVM and some other related locally linear classifiers. It is also comparable to a fine-tuned kernel SVM in terms of prediction performance, while it is more efficient than kernel SVM.

1 Introduction

In many problems of machine learning, the data points are distributed non-linearly. In this case, linear classifiers such as linear support vector machine [7] and linear regression [15] are not able to classify the data points correctly, because they fail to consider the underlying structures of complex data (e.g., clusters and manifolds). One way to solve this problem is to train a nonlinear classifier such as kernel support vector machine [23], which implicitly maps the input data into a high dimensional (or even infinite dimensional) feature space and learns a hyperplane in the new feature space to separate the mapped data. Unfortunately, although the training of linear SVM has received substantial advance in the past years [19] [25] [17] [16], the best known time complexity of training a kernel SVM is still quadratic to the number of examples [3]. Therefore, it is necessary to develop some classifiers which are able to handle the nonlinear data, while having considerably lower time complexity than nonlinear classifiers. In our study, we are interested in large margin classifiers, because they are based on max-margin principle, which is theoretically sound. They have also achieved great success in a wide range of applications.

In this paper, we propose a novel large margin classifier namely *Clustered Support Vector Machine (CSVM)*. In particular, we first divide the data into several clusters by K-means[1], and in each cluster, we train a linear support vector machine. To avoid over-fitting of each local SVM, we add a global regularization, which requires the weight vector of linear SVM in each cluster aligning with a global reference weight vector. The resulting CSVM can be efficiently solved by stochastic gradient descent [25], or dual coordinate descent [17]. We prove a data dependent generalization bound for CSVM based on Rademacher complexity [1], which theoretically explains why CSVM outperforms linear SVM. Experiments on benchmark large datasets show that the proposed method outperforms linear SVM and other related methods. Moreover, it achieves comparable or even better prediction performance than kernel SVM while it is computationally more efficient.

Appearing in Proceedings of the 16^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2013, Scottsdale, AZ, USA. Volume 31 of JMLR: W&CP 31. Copyright 2013 by the authors.

[1]The reason why we choose K-means is due to its well-known Vapnik-Chervonenkis dimension [26] [2], which is amenable to analysis. Other clustering algorithms may also be applicable.

It is worth noting that although we focus on large margin classifiers in this paper, the techniques developed here can be applied to other linear classifiers such as ridge regression and lasso [15] very straightforwardly.

The remainder of this paper is organized as follows. In Section 2, we briefly review some related work. In Section 3, we present clustered support vector machine, followed which we prove its generalization error bound. The experiments are demonstrated in Section 4. Finally, we draw conclusions and point out the future work in Section 5.

2 Related Work

There are quite a few existing studies which attempted to achieve the goal mentioned before from different perspectives.

The first way to handle the nonlinear data is local learning [4]. The basic idea of local learning is: given a testing example, we select a few training examples located in the vicinity of the testing example, and train a classifier with only those selected training examples, then apply this classifier to the testing example. Following this idea, there are several works with different heuristics, e.g., SVM-KNN [29] and fast local kernel machine (FaLKM) [24], etc. The disadvantage of local learning-based methods is their nature of lazy learning, which is inefficient during the testing phase, because they need to perform nearest neighbor searching and classifier training for every testing examples.

The second idea which has been widely explored is using a divide-and-conquer strategy, which is similar in spirit to the mixture of experts framework [18]. Typically, the input space is partitioned into disjoint clusters, followed which a local classifier is trained on examples falling in each cluster. This process is often repeated in the manner of expectation maximization (EM) [9]. Representative methods belonging to this family include [6] [30] [13] [8], to mention a few. The main disadvantage of mixture of experts-based methods is that they are prone to local minima, which makes model selection a nontrivial task. The proposed method in the present paper shares similar flavor with this family of methods. However, instead of performing clustering and training SVMs in an iterative way, we only perform it once in a global regularization way, which turns out to be very effective and efficient. We notice that there exist several studies [14] [22] which share similar spirit with ours, i.e., using global regularization (or ensemble) for a set of linear/nonlinear classifiers. However, they do not provide any non-trivial theoretical analysis.

Recently, there is a new family of methods which stems from the theory of locally linear approximation of nonlinear functions. The pioneering work is local coordinate coding (LCC) [28], which presented a principled coding algorithm with theoretically guarantee on the quality of the approximation. Based on local coordinate coding, [20] proposed a locally linear SVM (LLSVM), which approximates a nonlinear classifier by a set of linear classifiers associated with each anchor point in a dictionary. According to our empirical study, the main advantage of LLSVM is its superior prediction performance rather than its efficiency.

3 Clustered Support Vector Machine

In this section, we present clustered support vector machine, and analyze it theoretically. Throughout this paper, we consistently use lower case letters (e.g., w) to denote scalars, lower case bold letters to denote vectors (e.g., $\mathbf{w}$), upper case letters to denote the elements of a matrix (e.g., W), and bold-face upper case letters to denote matrices $\mathbf{W}$. $\mathbf{1}$ is a vector of all ones with an appropriate length, $\mathbf{0}$ is a vector of all zeros, and $\mathbf{I}$ is an identity matrix with an appropriate size. We use $\mathbf{w}^\top$ denote the transpose of a vector or a matrix. Furthermore, we use $\|\cdot\|$ denote the ℓ_2-norm of a vector.

3.1 The Proposed Model

Given a sample $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, we partition it into k clusters, i.e., $\{C_1, \ldots, C_k\}$ by some clustering algorithms such as K-means. Furthermore, we use $(\mathbf{x}_i^l, y_i^l), i = 1, \ldots, n_l$ to index the examples in the l-th cluster, where n_l is the number of examples in the l-th cluster. For each cluster, we are going to train a linear classifier, $f_l(\mathbf{x}), 1 \leq l \leq k$. The final classifier is defined with indicator function as follows

$$f(\mathbf{x}) = \sum_{l=1}^{k} f_l(\mathbf{x}) \mathbf{1}(\mathbf{x} \in C_l), \tag{1}$$

where $\mathbf{1}(\cdot)$ is an indicator function, and $f_l(\mathbf{x})$ is defined as

$$f_l(\mathbf{x}) = \mathbf{w}_l^\top \mathbf{x}. \tag{2}$$

Note that here we do not explicitly introduce a bias term b. This can be alternatively achieved by appending each example with an additional dimension: $\mathbf{x}^\top \leftarrow [\mathbf{x}^\top; 1]$ and $\mathbf{w}^\top \leftarrow [\mathbf{w}^\top; b]$.

The objective function of clustered support vector machine is as follows,

$$\begin{aligned}\arg\min_{\mathbf{w},\mathbf{w}_l,\xi_i^l\geq 0} \quad & \frac{\lambda}{2}\|\mathbf{w}\|^2+\frac{1}{2}\sum_{l=1}^{k}\|\mathbf{w}_l-\mathbf{w}\|^2 \\ & + \; C\sum_{l=1}^{k}\sum_{i=1}^{n_l}\xi_i^l, \\ \text{s.t.} \quad & y_i^l\mathbf{w}_l^\top\mathbf{x}_i^l\geq 1-\xi_i^l, i=1,\ldots,n_l,\forall l, \end{aligned} \tag{3}$$

where ξ_i^ls are slack variables, $\mathbf{w}$ is a global reference weight vector, $\frac{1}{2}\sum_{l=1}^{k}\|\mathbf{w}_l-\mathbf{w}\|^2$ is a global regularization, which requires the weight vector of each local linear SVM ($\mathbf{w}_l$) aligning with the global reference weight vector. $\mathbf{w}$ establishes a bridge among different clusters, such that the information from one cluster can be leveraged to another. It is therefore able to avoid over-fitting in each local cluster. In particular, if we set $\mathbf{w}=\mathbf{0}$, then CSVM will degenerate into k independent SVMs which are trained in each cluster separately.

Let $\mathbf{v}_l=\mathbf{w}_l-\mathbf{w}$, then $\mathbf{w}_l=\mathbf{v}_l+\mathbf{w}$. The above optimization problem can be equivalently written as

$$\begin{aligned}\arg\min_{\mathbf{w},\mathbf{v}_l,\xi_i^l\geq 0} \quad & \frac{\lambda}{2}\|\mathbf{w}\|^2+\frac{1}{2}\sum_{l=1}^{k}\|\mathbf{v}_l\|^2+C\sum_{l=1}^{k}\sum_{i=1}^{n_l}\xi_i^l, \\ \text{s.t.} \quad & y_i^l(\mathbf{v}_l+\mathbf{w})^\top\mathbf{x}_i^l\geq 1-\xi_i^l, i=1,\ldots,n_l,\forall l. \end{aligned} \tag{4}$$

In order to further simplify the above optimization problem, we define

$$\tilde{\mathbf{w}} = [\sqrt{\lambda}\mathbf{w}^\top,\mathbf{v}_1^\top,\ldots,\mathbf{v}_k^\top]^\top \tag{5}$$

$$\tilde{\mathbf{x}}_i^l = [\frac{1}{\sqrt{\lambda}}\mathbf{x}_i^{l\top},\mathbf{0}^\top\ldots,\mathbf{x}_i^{l\top}\ldots,\mathbf{0}^\top]^\top, \tag{6}$$

where the $(l+1)$-th component of $\tilde{\mathbf{x}}_i^l$ is $\mathbf{x}_i^l$.

Thus, the optimization problem in Eq. (4) can be simplified as

$$\begin{aligned}\arg\min_{\tilde{\mathbf{w}},\xi_i^l\geq 0} \quad & \frac{1}{2}\|\tilde{\mathbf{w}}\|^2+C\sum_{l=1}^{k}\sum_{i=1}^{n_l}\xi_i^l, \\ \text{s.t.} \quad & y_i^l\tilde{\mathbf{w}}^\top\tilde{\mathbf{x}}_i^l\geq 1-\xi_i^l, i=1,\ldots,n_l,\forall l. \end{aligned} \tag{7}$$

This is an appealing property because it indicates that CSVM can be efficiently solved by many off-the-shelf SVM packages such as Pegasus [25] and LibLinear [11], with time complexity $O(\frac{ndk}{\epsilon})$ or even $O(ndk\log(\frac{1}{\epsilon})$. Note that the complexity is linear to the number of clusters (k). In addition, the complexity of K-means is also linear to k. Therefore, the total training complexity is linear to the number of clusters, which makes CSVM applicable to large-scale datasets.

3.2 Relation to Regularized Multi-Task Learning

We notice that CSVM looks similar to regularized multi-task learning (RMTL) proposed in [10]. Using the terminology of multi-task learning, one may think that each cluster corresponds to a task, and the global regularization can be seen as the assumption that the hypotheses of each task share a common structure. In fact, CSVM is fundamentally different from RMTL. CSVM uses $\mathbf{w}_l^\top\mathbf{x}$ as the linear classifier for each cluster, while RMTL uses $\mathbf{v}_l^\top\mathbf{x}$ as the linear classifier for each task. More importantly, in CSVM, each cluster is a disjoint subset of the same sample (i.e., disjoint subsample) from a single distribution, while in RMTL, each task is an independent sample from possibly different distributions. Thus, the generalization analysis of CSVM is totally different from that of RMTL, which will be seen in the sequel.

3.3 Generalization Error Bound

In this subsection, we derive a generalization error bound for CSVM using the tool of Rademacher complexity for general function classes [1].

Definition 1. *[1] For a fixed sample set $S=\{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$ generated by a distribution $\mathcal{D}_\mathcal{X}$ on a set $\mathcal{X}$ and a real-valued function class $\mathcal{F}$ with domain $\mathcal{X}$, the empirical Rademacher complexity of $\mathcal{F}$ is the random variable*

$$\hat{R}_n(\mathcal{F})=\mathbb{E}_\sigma\left[\sup_{f\in\mathcal{F}}\frac{2}{n}\sum_{i=1}^{n}\sigma_i f(\mathbf{x}_i)\right], \tag{8}$$

where $\sigma=(\sigma_1,\ldots,\sigma_n)$ are independent uniform $\{\pm1\}$-valued random variables. The Rademacher complexity is

$$R_n(\mathcal{F})=\mathbb{E}_\mathbf{x}\left[\hat{R}_n(\mathcal{F})\right]. \tag{9}$$

Intuitively speaking, Rademacher complexity measures the richness of a class of real-valued functions with respect to a probability distribution. Note that here we use a stronger version of Rademacher complexity than that in [1], which omits an absolute value of the sum.

Let $\mathcal{G}$ be a class of binary-valued functions on $\mathcal{X}$, and let H be a class of real-valued functions on $\mathcal{X}$. In CSVM, $\mathcal{G}$ is the set of indicator functions of all the clusters generated by K-means. $\mathcal{H}$ is the set of candidate predictors used within each cluster, and assumed without loss of generality that $\sup_{\mathbf{x}\in\mathcal{X}}|h(\mathbf{x})|\leq 1$. In CSVM, $h_l(\mathbf{x})=\mathbf{w}_l^\top\mathbf{x}$. We say that a set of functions $g_1,\ldots,g_k\in\mathcal{G}$ is disjoint if, for any $\mathbf{x}\in\mathcal{X}$, $h_l(\mathbf{x})=1$ for only a single l. Then a cluster-wise predictor can be written as

$$f(\mathbf{x}) = \sum_{l=1}^{k} h_l(\mathbf{x}) g_l(\mathbf{x}), \tag{10}$$

where $h_1, \dots, h_k \in \mathcal{H}, g_1, \dots, g_k \in \mathcal{G}$ are disjoint.

For the ease of analysis, we define the following two composite function classes:

$$\begin{aligned} \mathcal{G}^k &= \{\mathbf{x} \to \mathbf{g}(\mathbf{x}) = [g_1(\mathbf{x}), \dots, g_k(\mathbf{x})]^\top | g_l \in \mathcal{G}\} \\ \mathcal{H}^k &= \{\mathbf{x} \to \mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), \dots, h_k(\mathbf{x})]^\top | h_l \in \mathcal{H}\}. \end{aligned} \tag{11}$$

Based on the above setup, the function class of CSVM can be defined as follows.

Definition 2. *The function class of CSVM is* $\mathcal{F} = \{\mathbf{x} \to \sum_{l=1}^k g_l(\mathbf{x}) \mathbf{w}_l^\top \mathbf{x} | \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{2}\sum_{l=1}^k \|\mathbf{w}_l - \mathbf{w}\|_2^2 \leq B, g_l \in \mathcal{G}\}$*, where* $B > 0$ *is a constant.*

It is easy to show that the optimal $\mathbf{w}$ to Eq. (3) is

$$\mathbf{w}^* = \frac{1}{\lambda + k} \sum_{l=1}^{k} \mathbf{w}_l. \tag{12}$$

Substitute Eq. (12) into the Definition 2, we obtain the following equivalent function class.

Definition 3. *The function class of CSVM is* $\mathcal{F} = \{\mathbf{x} \to \mathbf{g}^\top(\mathbf{x}) \mathbf{W}^\top \mathbf{x} | \frac{1}{2} tr(\mathbf{W}\mathbf{M}\mathbf{W}^\top) \leq B, \mathbf{g} \in \mathcal{G}^k\}$*, where* $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_k]$*,* $B > 0$ *is a constant, and* $\mathbf{M} = \frac{1}{\lambda+k}\mathbf{1}\mathbf{1}^\top + \frac{\lambda+k-4}{\lambda+k}\mathbf{I} + \mathbf{L}$ *with* $L_{lm} = -\frac{1}{\lambda+k}$ *if* $l \neq m$ *and* $L_{ll} = \frac{k-1}{\lambda+k}$*.*

In the following, we will present several theorems, which are among the main contributions of this paper.

Theorem 4. *The empirical Rademacher complexity of the function class for CSVM is upper bounded as below*

$$\hat{R}_n(\mathcal{F}) \leq \sup_{\mathbf{g} \in \mathcal{G}^k} \hat{R}_n(\mathcal{H}_{\mathbf{g}}^k) + 15k\sqrt{\frac{\log(n)}{n}}, \tag{13}$$

where $\mathcal{H}_{\mathbf{g}}^k$ *is an auxiliary function class defined as follows*

$$\mathcal{H}_{\mathbf{g}}^k = \{\mathbf{x} \to \langle \mathbf{h}(\mathbf{x}), \mathbf{g}(\mathbf{x}) \rangle : \mathbf{h} \in \mathcal{H}^k\} \tag{14}$$

for any fixed $\mathbf{g} \in \mathcal{G}^k$

Theorem 5. *The empirical Rademacher complexity of the auxiliary function class* $\mathcal{H}_{\mathbf{g}}^k$ *is upper bounded as*

$$\hat{R}_n(\mathcal{H}_{\mathbf{g}}^k) \leq \frac{3\sqrt{B}}{n} \sqrt{\sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i \mathbf{g}^\top(\mathbf{x}_i) \mathbf{M}^{-1} \mathbf{g}(\mathbf{x}_i)}, \tag{15}$$

The generalization error bound for CSVM is stated in the following theorem.

Theorem 6. *Fix* $\delta \in (0,1)$*, and let* $\mathcal{F}$ *be a class of functions mapping from* $\mathcal{X} \times \mathcal{Y}$ *to* $[0,1]$*. Let* $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ *be drawn independently according to a probability distribution* $\mathcal{D}$*. Then with probability* $1-\delta$ *over random draws of samples of size* n*, every* $f \in \mathcal{F}$ *satisfies*

$$\begin{aligned} err(f) &\leq \hat{err}(f) + \frac{3\sqrt{B}}{n} \sum_{l=1}^{k} \frac{\sqrt{n_l}}{n} \sqrt{\sum_{i=1}^{n_l} (\mathbf{x}_i^l) \cdot \mathbf{x}_i^l M_{ll}^{-1}} \\ &+ 15k\sqrt{\frac{\log(n)}{n}} + 3\sqrt{\frac{\ln(2/\delta)}{2n}}, \end{aligned} \tag{16}$$

where M_{ll}^{-1} *is the* l*-th diagonal element of* $\mathbf{M}^{-1}$*.*

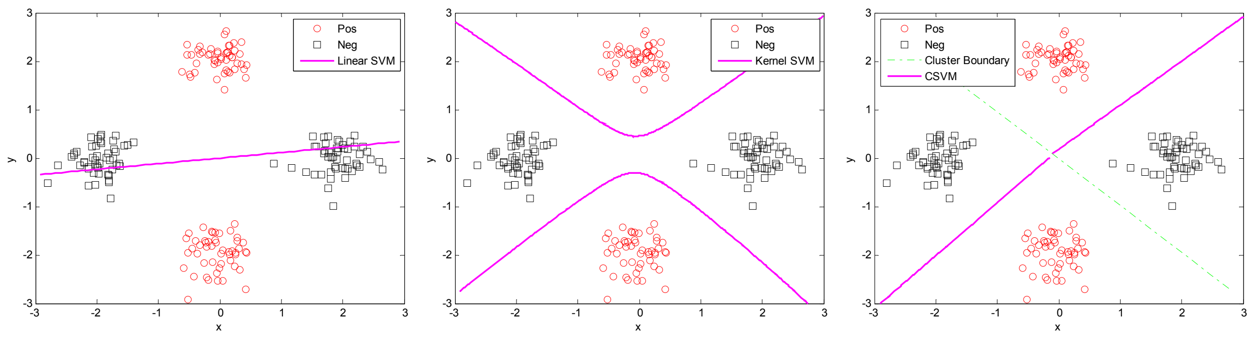
Recall that the empirical Rademacher complexity of linear SVM is $O\left(\frac{1}{n}\sqrt{\sum_{i=1}^n \mathbf{x}_i^\top \mathbf{x}_i}\right)$. The empirical Rademacher complexity of CSVM is $O\left(\frac{1}{n}\sum_{l=1}^k \frac{\sqrt{n_l}}{n}\sqrt{\sum_{i=1}^{n_l}(\mathbf{x}_i^l) \cdot \mathbf{x}_i^l M_{ll}^{-1}}\right)$, which is smaller than that of linear SVM when $k \geq 2$. Given that k grows slower than n, the third term in the right hand side of the above bound $15k\sqrt{\frac{\log(n)}{n}}$ is dominated by the empirical Rademacher complexity in the second term. In practice, we often choose $2 \leq k \ll n$.

4 Experiments

In this section, we evaluate the proposed method on both synthetic and real-world datasets, and compare it with linear SVM, kernel SVM, as well as other related methods discussed in Section 2.

4.1 Synthetic Dataset

To get an intuitive picture of how CSVM works for linearly nonseparable data, we first show the results of SVM, Kernel SVM and CSVM on a synthetic XOR dataset in Figure 1. This dataset contains four Gaussians: the top and the bottom ones constitute one class, while the other two constitute the other class. It is obvious that a global linear SVM is not able to correctly separate the data from the two classes. Kernel SVM is able to classify the data correctly by learning a nonlinear decision boundary. Our proposed CSVM partitions the data into two clusters (the green dashed curve is the boundary between two clusters), and train a linear SVM in each cluster (the pink lines are decision boundaries of linear SVMs). We can see that both KSVM and CSVM are able to sperate the linearly nonseparable data.



(a) Linear SVM (b) Kernel SVM (c) CSVM

Figure 1: Learned classifiers on the synthetic XOR dataset: (a) SVM; (b) KSVM; and (c) CSVM. The pink curves are the learned classifiers by different methods.

4.2 Real Datasets

We use four benchmark datasets: two of them are from LibSVM website[2], and the other two are digit recognition datasets which are originally used in [21] [12]. Note that all these datasets have already been split into training and testing sets. For each digit dataset, we transform it into a binary classification task by grouping the odd digits into one class and the even digits into another class. In addition, for MNIST data set, we also generate two relatively small binary classification tasks, which aim to distinguish 3 from 8, and 4 from 9. Table 1 summarizes the characteristics of these data sets.

Table 1: Description of the data sets

Datasets	#training	#test	#features	#classes
SVMGUIDE1	3,089	4000	4	2
IJCNN1	49,990	91,701	22	2
USPS OvE	7,291	917	256	2
MNIST 3v8	11,982	1984	784	2
MNIST 4v9	11,791	1991	784	2
MNIST OvE	60,000	10,000	784	2

For each example in every dataset, we normalize it into a vector with unit ℓ_2-norm.

4.3 Compared Methods and Parameter Settings

Linear SVM: The regularization parameter C is tuned by 5-fold cross validation on the training set by searching the grid $\{10^{-2}, 10^{-1}, 1, 10, 10^2\}$. We use the implementation of LibLinear [11].

Kernel SVM: We use Gaussian Kernel $K(\mathbf{x}, \mathbf{z}) = \exp(-\gamma\|\mathbf{x} - \mathbf{z}\|^2)$. The width of Gaussian kernel γ is tuned by 5-fold cross validation on the training set by searching the grid $\{10^{-2}, 10^{-1}, 1, 10, 10^2\}$. The regularization parameter C is tuned in the same way as above. We use the implementation in LibSVM [5].

SVM-KNN [29]: It is a representative method in the family of local learning. The number of nearest neighbor search is tuned by 5-fold cross validation on the training set via the grid $\{20, 50, 100\}$. The linear SVM for each testing example is tuned the same as before.

K-means+SVM: This is a simple baseline, which first partitions the dataset into k clusters, and then trains a linear SVM in each local cluster separately. Note that it can be seen as a special case of our proposed method when $\mathbf{w} = 0$. We set k equal to 8 empirically due to its good performance.

MSVM [13]: It is the state-of-the-art method in the category of mixture-of-experts. The EM algorithm is initialized with the same K-means clustering results of K-means+SVM, which makes the comparison as fair as possible. The number of clusters are set to 20 according to [13]. Note that the final number of clusters could be smaller than 20 due to the built-in model selection mechanism of the algorithm. The hyper-parameter of the prior (which corresponds to the regularization parameter C in standard SVM) is tuned the same as above. The scale parameter of the gating function is set to 1 according to the original paper.

LLSVM [20]: It is a representative method belonging to the family of local coordinate coding. For each dataset, the dictionary of anchor points is generated by K-means with $k = 100$ according to [20]. Given the generated dictionary, the coefficients of the local coordinate coding are obtained by the fast algorithm proposed in [27]. The parameters are tuned according to the description in that paper. The regularization parameter of LLSVM is tuned in the same way as linear SVM.

CSVM: The global regularization parameter λ is tuned by 5-fold cross validation on the training set by

[2]http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets /multilabel.html

the grid $\{1, 5, 10, 20, 50, 100\}$. The regularization parameter C is tuned the same as above. We set k equal to 8 the same as K-means+SVM. We also use the same K-means clustering results as K-means+SVM for fair comparison. We will study the sensitivity of CSVM with respect to k later.

For those algorithms which involve K-means clustering (e.g., K-means+SVM, MSVM, LLSVM and our method), due to the randomness of K-means clustering, we repeat the training process for 10 rounds, and compute the mean results (i.e., classification accuracy on the test set and training time) as well as their standard deviation. Note that when we calculate training time, we count the time of both K-means and SVM training, for the above methods.

For those algorithms which depend on nearest neighbor search (e.g., SVM-KNN, LLSVM), we use the data structure of k-d tree to speed up the search process. So the implementations of most baselines are optimized.

4.4 Results on Real Datasets

The comparison in terms of classification accuracy on the test set is shown in Table 2, while the comparison in terms of training time is shown in Table 3.

We observe that the proposed CSVM greatly outperforms linear SVM on all the datasets, which is consistent with our theory in Section 3.4. Furthermore, it is comparable to or even better than kernel SVM in terms of prediction accuracy (except on SVMGUIDE1 dataset, but we will see later that increasing the number of clusters improves its performance dramatically on this dataset), while its training is several orders of magnitude faster than kernel SVM.

In addition, we see that SVM-KNN often performs well and is comparable to kernel SVM. However, the training time of SVM-KNN is longer than CSVM, especially for large scale datasets (e.g., IJCNN1 and MNIST OvE). One may find that the training time of SVM-KNN is shorter than those reported elsewhere. The reason is that our implementation uses k-d tree to do nearest neighbor search, which is more efficient, especially for low-dimensional data.

The performance of LLSVM is even better than SVM-KNN, and is very close to our method. This implies the effectiveness of local coordinate coding. However, the training of LLSVM is slow, which sometimes takes more time than kernel SVM. The computational overhead of LLSVM includes nearest neighbor search and the local coordinate coding.

It is very interesting to see that the simple baseline, K-means+SVM, performs also very well. And it is also very efficient. However, K-means+SVM is worse than our method consistently. The reason is that it is lack of global regularization. Hence over-fitting in each cluster may happen.

MSVM is slightly better than K-means+SVM at most cases, which implies that the adjust of cluster assignment in an EM way could refine the local clusters. However, the resulting improvement in terms of prediction performance is quite limited, while its training time is at least one order of magnitude longer than that of K-means+SVM.

In a word, CSVM is a good alternative for kernel SVM. The superior performance of our method is attributed to its neat idea as well as its theoretical foundation, which guarantees small generalization error on the test data.

4.5 Study on the Number of Clusters k

Now we investigate how the performance of our method changes with respect to the number of clusters. We vary the number of clusters according to the grid $\{2, 5, 8, 11, 14, 17, 20\}$, and repeat the experiment of CSVM as in the previous subsection. The classification accuracy of CSVM with respect to various number of clusters is depicted in Figure 2, while the training time of our method with respect to varied number of clusters is illustrated in Figure 3. In all subfigures, the x-axis represents the number of clusters, while the y-axis is the averaged classification accuracy on the test data (in Figure 2) or training time (in Figure 3) over 10 runs. We also show the results of linear SVM for reference.

We can see that the classification accuracy of CSVM is relatively stable with different k on all the datasets except SVMGUIDE1. This is an appealing property because it implies that model selection would be easy. We attribute this property to the global regularization. This is consistent with our theoretical analysis in Section 3.4, which states that given $2 \leq k \ll n$, CSVM is stable with respect to k.

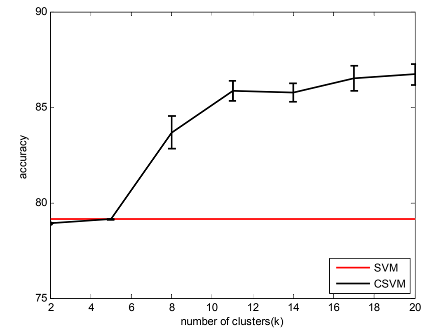
On the other hand, the training time of CSVM seems roughly linear in the number of clusters, this is what we expected by the analysis in Section 3.1. The curves fluctuate a bit because the convergence of K-means is a little sensitive to its initialization. Nevertheless, Figure 3 still gives a broad idea about the time consumption of CSVM with respect to different number of clusters. Since the prediction performance of CSVM is not sensitive to k, we could choose $k = 8$ or 11 in practice to get a tradeoff between effectiveness and efficiency.

Table 2: Comparison of classification accuracy (%) for different large margin classifiers

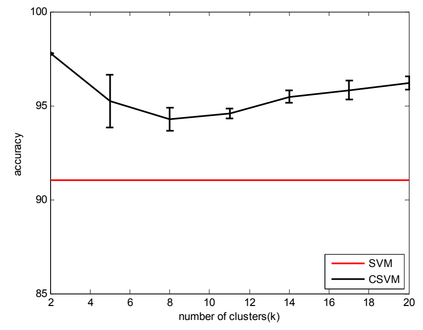
Dataset	SVMGUIDE1	IJCNN1	USPS OvE	MNIST 3v8	MNIST 4v9	MNIST OvE
Linear SVM	79.13	91.01	93.72	97.08	97.14	90.21
Kernel SVM	87.95	93.89	95.91	98.69	98.90	98.87
SVM-KNN	85.78	92.45	94.46	98.99	97.90	96.97
K-means+SVM	80.45±1.15	93.19±0.60	95.45±0.20	98.45±0.19	97.80±0.26	96.06±0.22
MSVM	83.27±0.64	93.41±0.19	95.71±0.22	98.46±0.25	97.74±0.25	96.62±0.18
LLSVM	87.64±0.30	94.07±0.45	96.19±0.25	98.62±0.18	98.09±0.24	98.91±0.09
CSVM	83.68±0.84	94.29±0.62	96.44±0.24	99.41±0.18	98.69±0.19	97.14±0.17

Table 3: Comparison of training time (in seconds) for different large margin classifiers

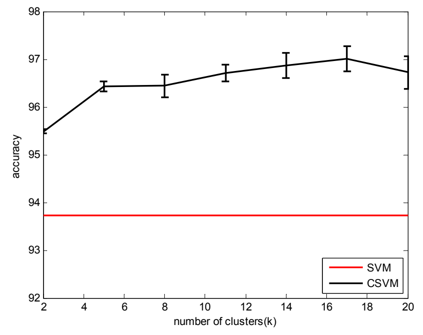
Dataset	SVMGUIDE1	IJCNN1	USPS OvE	MNIST 3v8	MNIST 4v9	MNIST OvE
Linear SVM	0.00	0.22	0.27	0.34	0.28	2.46
Kernel SVM	0.19	320.27	22.07	381.52	364.65	4031.03
SVM-KNN	2.62	164.97	16.57	38.78	37.42	1002.76
K-means+SVM	0.09±0.02	0.37±0.05	1.77±0.42	7.69±0.78	8.48±1.20	18.33±1.95
MSVM	2.09±0.01	10.56±0.05	42.86±0.55	192.64±1.13	199.15±1.12	440.28±2.80
LLSVM	2.17±0.08	472.88±1.67	73.09±3.00	188.04±4.10	151.24±5.38	1420.75±11.09
CSVM	0.32±0.06	4.13±1.33	3.60±0.57	12.25±1.28	10.81±1.57	31.93±3.37



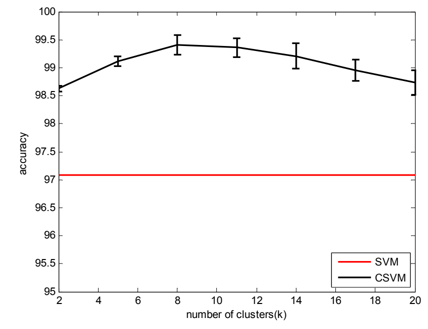
(a) SVMGUIDE1



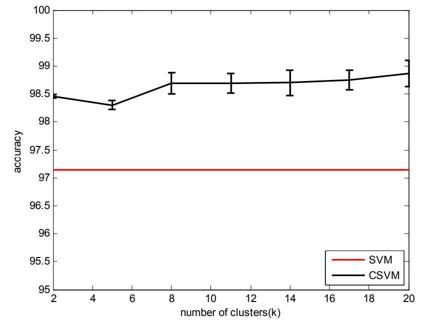
(b) IJCNN1



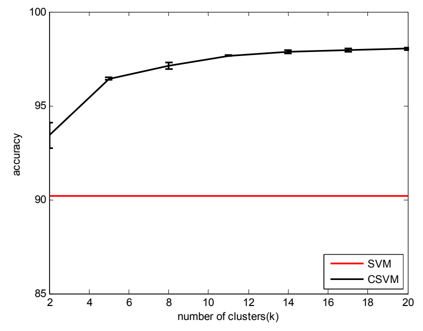
(c) USPS OvE



(d) MNIST 3v8



(e) MNIST 4v9

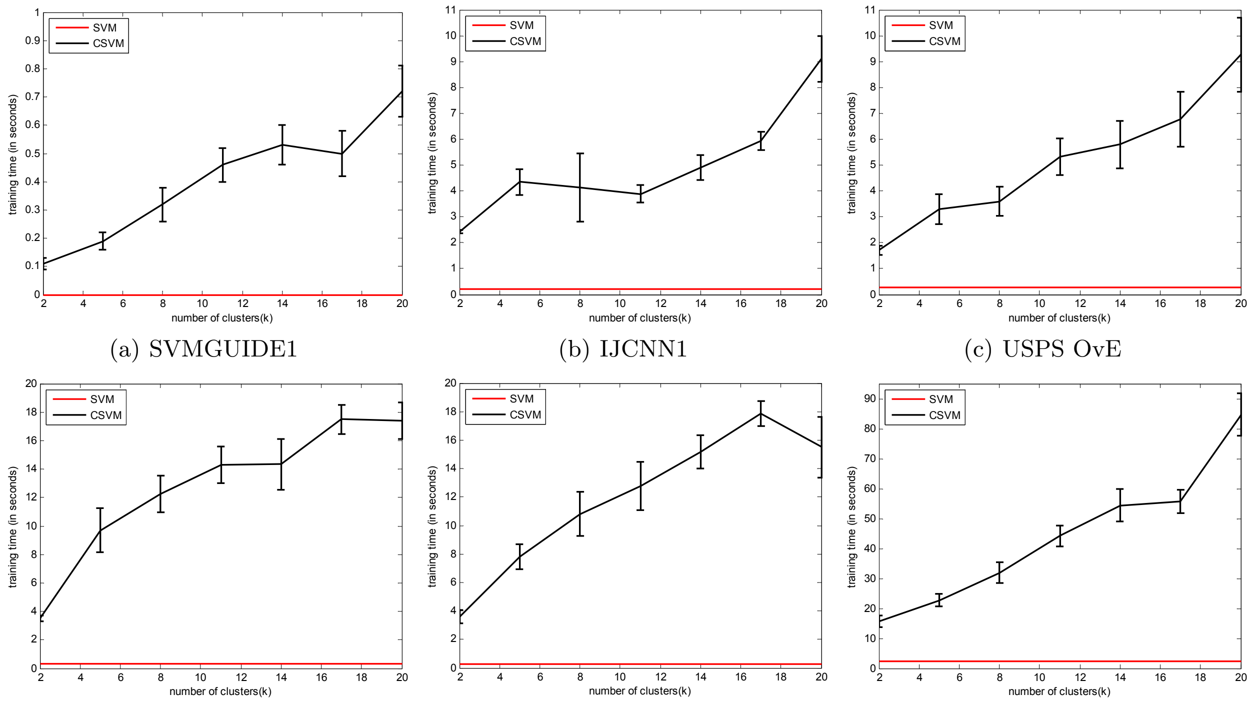


(f) MNIST OvE

Figure 2: Study on the classification accuracy of CSVM with respect to the number of clusters (k)

5 Conclusions and Future Work

In this paper, we proposed a *clustered support vector machine (CSVM)*, which divides the data into several clusters, followed which it trains a linear support vector machine in each cluster to separate the data locally. In the mean time, CSVM has an additional global regularization, which requires the weight vector of each local linear SVM aligning with a global weight vector. We show via experiments on several benchmark



(a) SVMGUIDE1 (b) IJCNN1 (c) USPS OvE

(d) MNIST 3v8 (e) MNIST 4v9 (f) MNIST OvE

Figure 3: Study on the training time of CSVM with respect to the number of clusters (k)

datasets that the proposed method outperforms linear SVM and some related locally linear classifiers. It is also comparable to a fine-tuned kernel SVM in terms of prediction performance, while it is more efficient than kernel SVM. In the future, we plan to extend the idea of clustered classifier to the setting of non-parametric classification/regression.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. The work was supported in part by U.S. National Science Foundation grants IIS-0905215, CNS-0931975, the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA), the U.S. Air Force Office of Scientific Research MURI award FA9550-08-1-0265, and MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC.

References

[1] P. L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

[2] S. Ben-David. A framework for statistical clustering with constant time approximation algorithms for *k*-median and *k*-means clustering. *Machine Learning*, 66(2-3):243–257, 2007.

[3] A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6:1579–1619, 2005.

[4] L. Bottou and V. Vapnik. Local learning algorithms. *Neural Computation*, 4(6):888–900, 1992.

[5] C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM TIST*, 2(3):27, 2011.

[6] R. Collobert, S. Bengio, and Y. Bengio. A parallel mixture of svms for very large scale problems. In *NIPS*, pages 633–640, 2001.

[7] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[8] O. Dekel and O. Shamir. There's a hole in my data space: Piecewise predictors for heterogeneous learning problems. *Journal of Machine Learning Research - Proceedings Track*, 22:291–298, 2012.

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[10] T. Evgeniou and M. Pontil. Regularized multi–task learning. In *KDD*, pages 109–117, 2004.

[11] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[12] M. A. Fanty and R. A. Cole. Spoken letter recognition. In *NIPS*, pages 220–226, 1990.

[13] Z. Fu, A. Robles-Kelly, and J. Zhou. Mixing linear svms for nonlinear classification. *IEEE Transactions on Neural Networks*, 21(12):1963–1975, 2010.

[14] B. Hamers, J. A. K. Suykens, and B. D. Moor. Ensemble learning of coupled parameterized kernel models. In *ICANN*, pages 130–133, 2003.

[15] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag, 2001.

[16] E. Hazan, T. Koren, and N. Srebro. Beating sgd: Learning svms in sublinear time. In *NIPS*, pages 1233–1241, 2011.

[17] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear svm. In *ICML*, pages 408–415, 2008.

[18] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, Mar. 1991.

[19] T. Joachims. Training linear svms in linear time. In *KDD*, pages 217–226, 2006.

[20] L. Ladicky and P. H. S. Torr. Locally linear support vector machines. In *ICML*, pages 985–992, 2011.

[21] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

[22] G. Qi, Q. Tian, and T. S. Huang. Locality-sensitive support vector machine by exploring local correlation and global regularization. In *CVPR*, pages 841–848, 2011.

[23] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

[24] N. Segata and E. Blanzieri. Fast and scalable local kernel machines. *Journal of Machine Learning Research*, 11:1883–1926, 2010.

[25] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *ICML*, pages 807–814, 2007.

[26] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[27] J. Wang, J. Yang, K. Yu, F. Lv, T. S. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR*, pages 3360–3367, 2010.

[28] K. Yu, T. Zhang, and Y. Gong. Nonlinear learning using local coordinate coding. In *NIPS*, pages 2223–2231, 2009.

[29] H. Zhang, A. C. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *CVPR (2)*, pages 2126–2136, 2006.

[30] J. Zhu, N. Chen, and E. P. Xing. Infinite svm: a dirichlet process mixture of large-margin kernel machines. In *ICML*, pages 617–624, 2011.