Linearithmic Time Sparse and Convex Maximum Margin Clustering

Xiao-Lei Zhang, *Student Member, IEEE*, and Ji Wu, *Member, IEEE*

***Abstract*—Recently, a new clustering method called maximum margin clustering (MMC) was proposed and has shown promising performances. It was originally formulated as a difficult nonconvex integer problem. To make the MMC problem practical, the researchers either relaxed the original MMC problem to inefficient convex optimization problems or reformulated it to nonconvex optimization problems, which sacrifice the convexity for efficiency. However, no approaches can both hold the convexity and be efficient. In this paper, a new linearithmic time sparse and convex MMC algorithm, called support-vector-regression-based MMC (SVR-MMC), is proposed. Generally, it first uses the SVR as the core of the MMC. Then, it is relaxed as a convex optimization problem, which is iteratively solved by the cutting-plane algorithm. Each cutting-plane subproblem is further decomposed to a serial supervised SVR problem by a new global extended-level method (GELM). Finally, each supervised SVR problem is solved in a linear time complexity by a new sparse-kernel SVR (SKSVR) algorithm. We further extend the SVR-MMC algorithm to the multiple-kernel clustering (MKC) problem and the multiclass MMC (M3C) problem, which are denoted as SVR-MKC and SVR-M3C, respectively. One key point of the algorithms is the utilization of the SVR. It can prevent the MMC and its extensions meeting an integer matrix programming problem. Another key point is the new SKSVR. It provides a linear time interface to the nonlinear kernel scenarios, so that the SVR-MMC and its extensions can keep a linearthmic time complexity in nonlinear kernel scenarios. Our experimental results on various real-world data sets demonstrate the effectiveness and the efficiency of the SVR-MMC and its two extensions. Moreover, the unsupervised application of the SVR-MKC to the voice activity detection (VAD) shows that the SVR-MKC can achieve good performances that are close to its supervised counterpart, meet the real-time demand of the VAD, and need no labeling for model training.**

***Index Terms*—Clustering, maximum margin, multiple-kernel learning (MKL), unsupervised learning, voice activity detection (VAD).**

I. Introduction

CLUSTERING finds a structure in a collection of unlabeled data and has been identified as a significant technique for many applications, such as classification of documents, marketing analysis, biology [1], human–computer interaction systems [2], and city planning. Since the early works in k-means clustering [3]–[5], data clustering has been studied for years, and many algorithms have been developed, such as mixture model [6], fuzzy clustering [7], [8], spectral clustering [9]–[11], graph-theoretic clustering [12], mean-shift clustering [13], [14], and agglomerative mean-shift clustering [15].

Recently, the maximum margin clustering (MMC) technique has attracted much attention [16]. It borrows the idea of the large margin from support vector machine (SVM) [17]. Unlike the support vector clustering [18]–[20], which aims at finding the smallest sphere in the feature space that encloses the images of the data and has a weak control on the number of clusters, MMC aims at finding not only the maximum margin hyperplane in the feature space but also the optimal label pattern, such that, if an SVM trained on the optimal label pattern, the optimal label pattern will yield the largest margin among all possible label patterns $\{\mathbf{y}|\mathbf{y}=\{y_i\}_{i=1}^n\}$, where n is the number of samples and y_i denotes the possible class of the ith sample. However, unlike supervised SVM, which is formulated as a convex optimization problem, the MMC is formulated as a nonconvex integer optimization problem, which is difficult to solve. Because of this, there are two research directions over MMC, i.e., how to relax the nonconvex integer MMC problem to a convex problem and how to decrease the time and storage complexities of the MMC.

Originally, Xu *et al.* [16] relaxed the integer label matrix $\mathbf{M}=\mathbf{y}\mathbf{y}^T$ in the dual form of the MMC to a semidefinite matrix and eventually reformulated the MMC problem to a convex semidefinite programming (SDP) problem. The experimental results showed that the MMC often achieved more accurate results than conventional clustering methods. Valizadegan and Jin [21] further proposed the generalized MMC (GMMC) method that reduces the number of parameters from $\mathcal{O}(n^2)$ in [16] to $\mathcal{O}(n)$. The GMMC has accelerated MMC by a factor of 100 times. However, the algorithms previously mentioned are formulated as SDP problems, which make them computationally intolerable when the data sets contain over hundreds of samples. As an example, for multiclass problems, the time complexity of the SDP-based method is as high as $\mathcal{O}(n^{6.5})$ [22].

In order to solve the MMC problem efficiently, several approaches sacrificed the convexity for efficiency [23]–[30]. For example, Zhang *et al.* [24], [25] proposed an alternative optimization algorithm, called iterative support vector regression (IterSVR), which converts the MMC problem to serial SVM training problems. Although the IterSVR has an empirical time complexity scaled between $\mathcal{O}(n)$ and $\mathcal{O}(n^{2.3})$, an auxiliary clustering algorithm is necessary for a good guess of its initial

Manuscript received July 7, 2011; revised January 10, 2012 and April 10, 2012; accepted April 27, 2012. Date of publication May 23, 2012; date of current version November 14, 2012. This work was supported in part by the National Natural Science Funds of China under Grant 61170197 and in part by the Planned Science and Technology Project of Tsinghua University under Grant 20111081023. This paper was recommended by Associate Editor L. Wang.

The authors are with the Multimedia Signal and Intelligent Information Processing Laboratory, Tsinghua National Laboratory for Information Science and Technology, Department of Electronic Engineering, Tsinghua University, Beijing 100084, China (e-mail: huoshan6@126.com; wuji_ee@tsinghua.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2012.2197824

label vector, and there is no guarantee on how fast it can converge. Zhao *et al.* [26] and Wang *et al.* [27] proposed a linear time cutting-plane MMC (CPMMC) algorithm. It employs the constrained concave–convex procedure (CCCP) [31], [32] to decompose the MMC problem into serial supervised SVM problems, and it uses the linear time cutting-plane SVM solver [33], [34] to solve each SVM problem. Because the CCCP is a nonconvex optimization tool, the CPMMC algorithm suffers from local minima. Moreover, its linear time complexity is restricted to the linear kernel; the nonlinear kernel is accessed by kernel decompositions, such as the kernel principle components analysis (KPCA) [35], [36] or the Cholesky decomposition [37], which results in an additional time complexity of at least $\mathcal{O}(n^2)$ [38], [39]. This weakness is rather explicit in its extension to the unsupervised multiple-kernel learning (MKL), called multiple-kernel clustering (MKC), which mainly works in nonlinear kernel-induced feature spaces [40].

To avoid nonconvexity and SDP simultaneously, more recently, Li *et al.* [41] proposed the convex label-generating MMC (LG-MMC), which has time and storage complexities of $\mathcal{O}(n^2)$. It first relaxes the MMC problem by constructing a convex hull [42] on all feasible label matrices $\mathbf{M}$ and then solves the relaxed MMC problem approximately via the cutting-plane algorithm (CPA) [33]. Not only the LG-MMC is much faster than its previous convex relaxation methods [16], [21], but also, the convex relaxation method of the LG-MMC, which is to construct a convex hull [42] on all feasible label matrices $\mathbf{M}$, $\mathbf{M} = \mathbf{y}\mathbf{y}^T$, is the tightest convex relaxation of $\mathbf{M}$. However, LG-MMC still seems time consuming on large-scale data sets (as shown in Section IX). Aside from the utilization of the simple MKL solver [43], constructing a convex hull on the integer matrix $\mathbf{M}$ hinder the LG-MMC from utilizing efficient SVM solvers to further lower the overall time complexity.

In this paper, we propose a new MMC algorithm called SVR-MMC. Specifically, we use the Laplacian-loss SVR [44], [45] as the core of the MMC. Then, we relax the SVR-based MMC problem to a convex optimization problem by constructing a convex hull on the integer label vector $\mathbf{y}$ of the new objective, which is a similar convex formulation method with the LG-MMC in [41]. The relaxed problem is iteratively solved by CPA. Moreover, in each cutting-plane iteration, a single cutting-plane subproblem is further decomposed to a serial supervised SVR learning problems by a new global extended-level method (GELM). At last, we apply the cutting-plane subspace pursuit (CPSP) algorithm [33], [34], [46], [47], which is a combination of the CPA and the sparse-kernel estimation techniques [38], [48], to solve each SVR problem.

For real-world applications, we also extend the SVR-MMC to the MKC scenario and the multiclass MMC (M3C) scenario, which are denoted as SVR-MKC and SVR-M3C, respectively.

Technically, the following two items are important for the advantages of the proposed algorithms.

1) Using the SVR [44], [45] as the core prevents the MMC and its extensions meeting an integer matrix programming problem. Therefore, we can utilize efficient supervised SVM techniques to lower the time and storage complexities of the proposed algorithms.
2) Using the sparse-kernel estimation techniques [38], [48] makes the MMC and its extensions work in linear time complexities with nonlinear kernels. This utilization is rather important. As we know, the discriminability of large margin methods can be greatly improved in the framework of kernel learning.

The main contributions of this paper are as follows.

1) The SVR-MMC is both effective and efficient. From the respect of the effectiveness, the SVR-MMC is formulated as a convex one, such that a global minimum solution is available. From the respect of the efficiency, the following are considered: 1) In the linear kernel scenario, the SVR-MMC has the lowest time complexity among the convex MMC algorithms [16], [21], [41]. It also achieves a comparable time complexity with the fastest nonconvex MMC algorithm [26]. 2) In the nonlinear kernel scenarios, the SVR-MMC is the most efficient one in existing MMC algorithms.
2) The SVR-MKC and the SVR-M3C maintain all properties and advantages of the SVR-MMC.
   a) The SVR-MKC is not only formulated as a convex one but also much faster than the existing nonconvex MKC algorithm [40].
   b) The SVR-M3C is a convex M3C method. It is much faster than its previous convex M3C algorithm [22], which scales with $\mathcal{O}(n^{6.5})$ in time. It achieves a comparable time complexity with the nonconvex M3C [27], [49] in the linear kernel scenario. It is the most efficient one in the nonlinear kernel scenarios.

The rest of this paper is organized as follows: In Section III, we briefly introduce some works related to this paper. In Section IV, we present a new SVR-MMC algorithm. In Section V, we present the multiple-kernel extension of the SVR-MMC, which is called SVR-MKC. In Section V, we present the multiclass extension, i.e., SVR-M3C. In Section VII, we explain in brief why we use SVR. In Section VIII, we analyze the time and storage complexities of the proposed algorithms briefly. In Section IX, we report the experimental results on various data sets and further apply the SVR-MKC algorithm to the voice activity detection (VAD). Finally, we conclude this paper in Section X.

II. Notations

We first introduce some notations here. Bold small letters, e.g., $\mathbf{w}$ and $\boldsymbol{\alpha}$, indicate vectors. Bold capital letters, e.g., $\mathbf{K}$ indicate matrices. Letters in calligraphic bold fonts, e.g., $\mathcal{A}$, $\mathcal{B}$, $\mathcal{Y}$, and $\mathbb{R}$, indicate sets, where $\mathbb{R}^n$ denotes an n-dimensional real space. $\mathbf{0}$ ($\mathbf{1}$) is a vector with all entries being 1 (0). Operator ^{T} denotes the transpose. Operator $\circ$ denotes the element-wise product, and $\langle \mathbf{x}, \mathbf{y} \rangle$ defines the inner product of $\mathbf{x}$ and $\mathbf{y}$. Operator $\| \cdot \|^m$ denotes the m-norm, where m is a constant. The abbreviation "s.t." is short for "subject to." $E(\boldsymbol{\alpha}; \boldsymbol{\beta})$ denotes function E with parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.

III. Related Works

A. CPA

Recently, the CPA [33] has been employed into SVM, such as SVM$^{\mathrm{perf}}$ [34], [50], [51] and bundle method based risk minimization [46]. It lowers the complexity of large-scale SVM problems significantly. In CPA terminology, a problem with a full constraint set is called a master problem [47], whereas a problem with only a constraint subset from the full set is called a reduced problem or a cutting-plane subproblem.

Generally, CPA begins with a reduced problem that only has an empty working constraint set and then iterates two steps.

1) Solve the reduced problem.
2) Add one most violated constraint at the current solution point from the full set to the working constraint set, so as to form a new reduced problem.

If the newly generated constraint violates the solution of the reduced problem by no more than ϵ, CPA can be stopped, where ϵ is a user-defined cutting-plane solution precision. It has been proved that the number of the iterations scales only with $\mathcal{O}(1/\epsilon)$ [46].

B. MMC

MMC is to extend the theory of the supervised SVM to the unsupervised learning scenario. Given the unlabeled samples $\{\mathbf{x}_i\}_{i=1}^n$ with $\mathbf{x}_i \subseteq \mathbb{R}^d$, MMC aims to find their best labels $\mathbf{y} = \{y_i\}_{i=1}^n$, such that an SVM trained on $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ will yield the largest margin. It can be formulated as the following computational optimization problem:

$$\min_{\mathbf{y}\in\mathcal{B}_0} \min_{\mathbf{w},b} \frac{1}{2}|\mathbf{w}|^2 + C\ell\left(\{f_{\mathbf{w},b}(\mathbf{x}_i)\}_{i=1}^n, \mathbf{y}\right) \tag{1}$$

where set $\mathcal{B}_0 = \{\mathbf{y}|y_i \in \{\pm 1\}, -l \le \mathbf{1}^T\mathbf{y} \le l\}$ with $l \ge 0$ being a constant and ℓ is the empirical risk of function f. f takes a linear form with hyperplane parameters $\mathbf{w}$ and b as $f_{\mathbf{w},b}(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) + b$, where $\phi(\cdot)$ is the mapping function that is to map $\mathbf{x}_i$ into a (possibly) high-dimensional kernel-induced feature space. Constraint $-l \le \mathbf{1}^T\mathbf{y} \le l$ in $\mathcal{B}_0$ was added in [16] to avoid classifying all samples to only one class with a very large margin.

Unfortunately, unlike the supervised SVM, the MMC was formulated as a difficult nonconvex mixed-integer-programming (MIP) problem. Researchers have tried several approaches to reformulate this problem for its practical use. However, these approaches either reformulated MMC as nonconvex optimization problems or relaxed it as inefficient convex optimization problems. For the nonconvex formulation methods [23]–[28], [30], the CPMMC [26], [27] has linear time and storage complexities with the linear kernel. For the convex relaxation methods [16], [21], [41], the LG-MMC is the tightest convex relaxation of the original MMC problem and has time and storage complexities of $\mathcal{O}(n^2)$.

Summarizing the aforementioned, no method can both maintain the convexity and have a linear time complexity. In this paper, we will try to solve this problem.

C. MKL

The success of the kernel methods, such as SVM and MMC, strongly relies on a good selection of the kernel representations/functions/models of data samples. The kernel function is specified by the inner product of data points mapped in a kernel-induced feature space, e.g., $K(\cdot,\cdot) = \langle\phi(\cdot), \phi(\cdot)\rangle$. Recently, the problem of learning the optimal kernel functions has attracted much attention. One of the essential kernel learning methods is the MKL [43], [52]–[57].

Given Q mapping functions $\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_Q(\mathbf{x})$ corresponding with Q base kernel functions $K_1, \ldots, K_Q$, the MKL tries to find an optimal linear combination of the multiple predefined kernel functions K_q, $q = 1, \ldots, Q$, (referred to as base kernel functions) by minimizing the following nonconvex optimization problem:

$$\min_{\mathbf{w},b,\boldsymbol{\theta}\ge\mathbf{0}} \frac{1}{2}\|\mathbf{w}\|^2 + C\ell\left(\{f_{\mathbf{w},b,\boldsymbol{\theta}}(\mathbf{x}_i)\}_{i=1}^n, \mathbf{y}\right) \qquad \text{s.t. } J(\boldsymbol{\theta}) \le 1 \tag{2}$$

where ℓ is the empirical risk of function f and ℓ is supposed to be a convex function, $\boldsymbol{\theta} = \{\theta_q\}_{q=1}^Q$ is the kernel weight vector that needs to be learned from the samples, $J(\boldsymbol{\theta}) \le 1$ is the convex constraint on $\boldsymbol{\theta}$, and $f_{\mathbf{w},b,\boldsymbol{\theta}}(\mathbf{x})$ is defined as

$$f_{\mathbf{w},b,\boldsymbol{\theta}} = \mathbf{w}^T\phi_{\boldsymbol{\theta}}(\mathbf{x}) + b = \sum_{q=1}^Q \sqrt{\theta_q}\mathbf{w}_q^T\phi_q(\mathbf{x}) + b \tag{3}$$

where $(\mathbf{w}, b)$ are the hyperplane parameters of the MKL, weight $\mathbf{w}$ is defined as $\mathbf{w} = [\mathbf{w}_1^T, \ldots, \mathbf{w}_Q^T]^T$, and the composite feature mapping $\phi_{\boldsymbol{\theta}}(\cdot)$ is defined as $\phi_{\boldsymbol{\theta}}(\cdot) = [\sqrt{\theta_1}\phi_1(\cdot)^T, \ldots, \sqrt{\theta_Q}\phi_Q(\cdot)^T]^T$.

As the authors in [55] and [58] did, the nonconvexity of (2) can be avoided by applying the variable transformation $\mathbf{v}_q = \sqrt{\theta_q}\mathbf{w}_q$. Hence, problem (2) can be reformulated as

$$\min_{\mathbf{v},b,\boldsymbol{\theta}\ge\mathbf{0}} \frac{1}{2}\sum_{q=1}^Q \frac{\|\mathbf{v}_q\|^2}{\theta_q} + C\ell\left(\{f_{\mathbf{v},b}(\mathbf{x}_i)\}_{i=1}^n, \mathbf{y}\right) \qquad \text{s.t. } J(\boldsymbol{\theta}) \le 1 \tag{4}$$

where $\mathbf{v} = [\mathbf{v}_1^T, \ldots, \mathbf{v}_Q^T]^T$ and $f_{\mathbf{v},b}(\mathbf{x})$ is defined as

$$f_{\mathbf{v},b}(\mathbf{x}) = \sum_{q=1}^Q \mathbf{v}_q\phi_q(\mathbf{x}) + b. \tag{5}$$

Another challenge work is on the computation of MKL. Due to the efficiency in solving SVM, a number of SVM-based techniques have been proposed to alleviate the computational burden of MKL. Examples include sequential minimal optimization [53], quadratically constrained quadratic programming [55], semi-infinite linear programming [54], gradient method [43], [59], and extended level method (ELM) [56], [57]. Many of the MKL techniques belong to the wrapping-based methods. In the wrapping-based methods, the first step is to search for the optimal $\mathbf{v}$, b by an SVM solver given $\boldsymbol{\theta}$. The second step is to renew $\boldsymbol{\theta}$ to further decrease the objective value of the MKL given $\mathbf{v}$ and b.

However, traditional MKL researches are mostly focused on the supervised machine learning scenario; the MKL problem in the unsupervised learning scenario is an insufficiently explored research topic yet. Only recently, Zhao *et al.* [40] proposed an unsupervised MKL algorithm, called cutting-plane MKC (CPMKC). The CPMKC is a multiple-kernel extension of the MMC. It employs the CCCP [31], [32] to decompose the MKC problem to a serial convex second order cone programming (SOCP) problems, which can be solved in a similar way with [53] in polynomial time. The experimental results showed that the CPMKC algorithm can lead to better clustering results than the single-kernel-based CPMMC algorithm [26], [27] and can moreover emphasize those most powerful kernel functions in a given application. However, because CCCP is a nonconvex optimization tool, the CPMKC algorithm suffers from local minima. In the respect of efficiency, current SOCP solvers are still too slow to meet the command of large-scale problems, and the CPMKC algorithm has to first calculate each full base kernel matrix in time $\mathcal{O}(n^2)$. Then, it gets explicit expressions of the data samples in each kernel-induced feature space by some expensive kernel matrix decomposition methods, such as KPCA [35].

Summarizing the aforementioned, the CPMKC suffers from local minima and seems still inefficient in dealing with large-scale problems.

1) ELM: Here, we introduce a state-of-the-art MKL method that will be later used in our proposed methods.

The level method [60] is from the family of bundle methods. It has recently been extended to efficiently solve MKL problems [56], [57]. However, the core idea of the ELM is irrelevant to the MKL problem itself. Given a concave–convex objective function $E(\boldsymbol{\theta};\boldsymbol{\alpha})$ that is convex on $\boldsymbol{\theta}$ and concave on $\boldsymbol{\alpha}$, the ELM is to provide a serial increasingly tight (upper and lower) bounds for the optimal objective value $E(\boldsymbol{\theta}^\star;\boldsymbol{\alpha}^\star)$, where $(\boldsymbol{\theta}^\star,\boldsymbol{\alpha}^\star)$ denotes the optimal solution. Generally speaking, it iterates the following two steps until convergence.

For the first step, it solves the concave problem $\max_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^j;\boldsymbol{\alpha})$ with known $\boldsymbol{\theta}^j$ so as to get $\boldsymbol{\alpha}^j$, where j denotes the current ELM iteration number.

For the second step, it aims to obtain $\boldsymbol{\theta}^{j+1}$ by projecting $\boldsymbol{\theta}^j$ into a level set $\mathcal{L}_j$ that is constructed from the last j solutions $\{(\boldsymbol{\theta}^i,\boldsymbol{\alpha}^i)\}_{i=1}^{j}$. Specifically, for any optimal solution $(\boldsymbol{\theta}^\star,\boldsymbol{\alpha}^\star)$, we have the fact that $E(\boldsymbol{\theta}^\star;\boldsymbol{\alpha}) \le \max_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^\star;\boldsymbol{\alpha}) = E(\boldsymbol{\theta}^\star;\boldsymbol{\alpha}^\star) = \min_\theta E(\boldsymbol{\theta};\boldsymbol{\alpha}^\star) \le E(\boldsymbol{\theta};\boldsymbol{\alpha}^\star)$. By utilizing the inequality, we can find the following lower bound $\underline{E}^j$ and upper bound $\overline{E}^j$ of the objective $E(\boldsymbol{\theta};\boldsymbol{\alpha})$:

$$\underline{E}^j = \min_\theta h^j(\boldsymbol{\theta}) \qquad \overline{E}^j = \min_{1\le i\le j} E(\boldsymbol{\theta}^i;\boldsymbol{\alpha}^i)$$

where $h^j(\boldsymbol{\theta}) = \max_{1\le i\le j} E(\boldsymbol{\theta};\boldsymbol{\alpha}^i)$ is a cutting-plane model. Then, a level set, which specifies the set of solutions where the objective $E(\boldsymbol{\theta};\boldsymbol{\alpha})$ is bounded by $\underline{E}^j$ and $\overline{E}^j$, can be defined as $\mathcal{L}^j = \{\boldsymbol{\theta}|h^j(\boldsymbol{\theta}) \le V^j = \tau\overline{E}^j + (1-\tau)\underline{E}^j\}$. Finally, $\boldsymbol{\theta}^{j+1}$ is obtained by projecting $\boldsymbol{\theta}^j$ into the level set $\mathcal{L}^j$, which is formulated as the following optimization problem:

$$\min_{\theta^{j+1}} \|\boldsymbol{\theta}^{j+1} - \boldsymbol{\theta}^j\|^2 \quad \text{s.t. } E(\boldsymbol{\theta}^{j+1};\boldsymbol{\alpha}^i) \le V^j \qquad \forall i=1,\ldots,j.$$

We call the construction of the level set and the projection process as the *projection* function.

D. Multiclass Classification

Multiclass classifiers are more useful than binary class classifiers in our real-world life. Also, multiclass classification problem is a very broad research area. In this paper, we focus on the large margin related topics.

The SVM was originally proposed to tackle binary-class problems only [17]. To make it available for a multiclass problem, the error-correcting output code (ECOC) was introduced [61]. It decomposes the multiclass problem to a serial binary-class problem and then manipulates on the outputs of the dichotomizers. The ECOC belongs to the category of the classifier ensembles [62]–[66]. Generally, the ECOC has two research directions. One is called the *problem-independent* coding design, which tries to find a code set that has a good error-correcting ability without taking the characteristics of the data set into consideration. The common one-versus-one and one-versus-all codes belong to this class [67]. The other is called the *problem-dependent* coding design, which tries to find a code set that dedicates to a given data set [68]–[73].

One of the well-known problem-dependent coding design is the multiclass SVM that is proposed in [69]. In the beginning, they try to find the optimal problem-dependent binary codes, which results in a MIP problem. To avoid MIP, they relaxed the binary codes to continuous codes. Finally, they formulate a "multiclass SVM" objective, which can be solved in time $\mathcal{O}(n^2)$. There are also several similar works that try to solve multiclass problems in a single objective with large margin thoughts [74]–[80].

Compared with the ECOC-based multiclass SVM [69], the multiclass SVR approaches seem more natural for multiclass problems [79], [80]. They just assign each class a unique codeword and try to find the classification parameters that are the most suitable ones to the codewords [80].

For the unsupervised multiclass SVM, which is also known as M3C, it still has no uniform solution framework. Because label y is generalized from a binary integer value, e.g., $\{-1,+1\}$, to a multiple integer value, the M3C problem seems more complicated than the binary-class MMC problem. Currently, there are only two M3C techniques. The first M3C method is a convex one that is based on SDP [22]. It has a time complexity as high as $\mathcal{O}(n^{6.5})$. The second one is a nonconvex one, named cutting-plane M3C (CPM3C) [49]. The CPM3C is solved in the framework of CCCP in linear time with the linear kernel, but it suffers from local minima and is inefficient with nonlinear kernels.

Here comes the question. Can we solve M3C problem efficiently with a global optimum solution?

E. Overview of the Related Work

Our SVR-MMC can be seen as a technical combination of CPA [33], ELM [56], [57], LG-MMC [41], and the single kernel SVM$^{\text{perf}}$ [34], [50], [51] solver. The CPA provides an efficient framework of solving a problem that has a large number of constraints. The ELM provides an efficient wrapping-based method for the concave–convex problem that our MMC will meet. The LG-MMC provides a convex relaxation method for the MIP problem that our MMC will meet. The SVM$^{\text{perf}}$ solver provides several efficient algorithms for our SVR subproblem. It also provides linear time interfaces to nonlinear kernels, but all the aforementioned techniques have been revised for our special task.

Our SVR-MKC and SVR-M3C are natural extensions of the SVR-MMC, with the SVR replaced by the multiple-kernel SVR and the multiclass SVR, respectively.

IV. SVR-Based MMC

In this section, we will first present an SVR-based MMC objective, which will be relaxed as a concave–convex optimization problem. Then, we will solve the relaxed problem by the CPA. Each cutting-plane subproblem is further decomposed to a serial supervised SVR problem by a new GELM. Finally, each SVR problem will be solved by the CPSP algorithm.

A. Objective Formulation

As analyzed in Section VII, existing convex MMC algorithms operate on the label matrix $\mathbf{M} = \mathbf{y}\mathbf{y}^T$. The difficulty with $\mathbf{M}$ lies in the fact that we have to either minimize the objective function with $\mathcal{O}(n^2)$ parameters [16] or do a serial heavy computations related to $\mathbf{M}$ [41]. Therefore, we should redefine the MMC objective so as to avoid operating on $\mathbf{M}$. The SVR provides us this feasibility.

Originally, given a training set of n examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ with $y_i \in \mathbb{R}$, the SVR is used to estimate the linear regression $f(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x})$ with precision ε [81], [82].[1] For this, we minimize[2]

$$\frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^n |y_i - f(\mathbf{x}_i)|_\varepsilon$$

where $|\cdot|_\varepsilon$ is the ε-insensitive loss function defined as

$$|z|_\varepsilon = \max\{0, |z| - \varepsilon\}.$$

Written as a constraint optimization problem, it amounts to the following problem:

$$\begin{aligned}\min_{\mathbf{w},\xi_i\geq 0,\xi_i^*\geq 0} & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^n (\xi_i + \xi_i^*)\\ \text{s.t. } & y_i - \mathbf{w}^T\phi(\mathbf{x}_i) \leq \varepsilon + \xi_i,\\ & -y_i + \mathbf{w}^T\phi(\mathbf{x}_i)\, leq \varepsilon + \xi_i^* \quad \forall i = 1,\ldots,n.\end{aligned}$$

If we use SVR as a binary-class classifier, we just need to restrict y_i to $\{-1, 1\}$.

In this paper, we focus on the SVR with $\varepsilon = 0$, which is the Laplacian-loss SVR [44], [45], i.e.,

$$\begin{aligned}\min_{\mathbf{w},\xi_i\geq 0,\xi_i^*\geq 0} & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^n (\xi_i + \xi_i^*)\\ \text{s.t. } & y_i - \mathbf{w}^T\phi(\mathbf{x}_i) \leq \xi_i,\\ & -y_i + \mathbf{w}^T\phi(\mathbf{x}_i) \leq \xi_i^* \quad \forall i = 1,\ldots,n.\end{aligned} \tag{6}$$

[1]The bias term b is not considered.

[2]To better capture the scaling behavior of C, we divide it by n.

The empirical risk of the SVR is $\ell_l = (1/n)\sum_{i=1}^n(\xi_i + \xi_i^*) = (1/n)\sum_i |y_i - \mathbf{w}^T\phi(\mathbf{x}_i)| = (1/n)\sum_i |1 - y_i\mathbf{w}^T\phi(\mathbf{x}_i)|$, which is similar to the squared hinge loss $\ell_s = \sum_{i=1}^n \xi_i^2 = \sum_i(1 - y_i\mathbf{w}^T\phi(\mathbf{x}_i))^2$ in [41].

Here, we use the Laplacian-loss SVR as the core of the MMC problem. The new MMC problem is formulated as follows:

$$\min_{\mathbf{y}\in\mathcal{B}_0}\left\{\begin{aligned}\min_{\mathbf{w},\xi_i\geq 0,\xi_i^*\geq 0} & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^n(\xi_i+\xi_i^*)\\ \text{s.t. } & y_i - \mathbf{w}^T\phi(\mathbf{x}_i) \leq \xi_i,\\ & -y_i + \mathbf{w}^T\phi(\mathbf{x}_i) \leq \xi_i^*, \quad \forall i = 1,\ldots,n\end{aligned}\right\}. \tag{7}$$

1) One-Slack Formulation: Objective (7) has $2n$ unknown slack variables. Inspired by the formulation method of the efficient SVM$^{\text{perf}}$ solver [34], [50], [51], now, we will reduce the number of slack variables from $2n$ to 1 by reformulating problem (7) into the following optimization problem:

$$\min_{\mathbf{y}\in\mathcal{B}_0}\left\{\begin{aligned}&\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C\xi\\ &\text{s.t. } \frac{1}{n}\sum_{i=1}^n g_i y_i - \frac{1}{n}\sum_{i=1}^n g_i\mathbf{w}^T\phi(\mathbf{x}_i) \leq \xi, \forall \mathbf{g} \in \{1,-1\}^n\end{aligned}\right\}. \tag{8}$$

Problems (7) and (8) are equivalent in the following theorem.

Theorem 1: Any solution $(\mathbf{w}, \mathbf{y})$ of problem (8) is also a solution of problem (7) and *vice versa*, with $\xi = (1/n)\sum_{i=1}^n(\xi_i + \xi_i^*)$.

Proof: The proof of Theorem 1 is provided in Appendix A. ■

We are to solve (8) in the dual of the SVR in the brackets as follows:

$$\min_{\mathbf{y}\in\mathcal{B}_0}\max_{\boldsymbol{\alpha}\in\mathcal{A}} E(\mathbf{y};\boldsymbol{\alpha}) \tag{9}$$

where the semicolon ";" is used to separate different parameters, $\boldsymbol{\alpha} = \{\alpha_i\}_{i=1}^{2^n}$ is a dual variable vector with 2^n elements, $\mathcal{A} = \{\boldsymbol{\alpha}|\boldsymbol{\alpha}^T\mathbf{1} \leq C, \boldsymbol{\alpha} \geq \mathbf{0}\}$, and $E(\mathbf{y};\boldsymbol{\alpha})$ is defined as

$$E(\mathbf{y};\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T\mathbf{G}^T\mathbf{y} - \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{G}^T\mathbf{K}\mathbf{G}\boldsymbol{\alpha}. \tag{10}$$

We can also get $\mathbf{w}$ as

$$\mathbf{w} = \sum_{k=1}^{2^n}\alpha_k\left(\frac{1}{n}\sum_{i=1}^n g_{k,i}\phi(\mathbf{x}_i)\right). \tag{11}$$

B. Convex Relaxation

Problem (9) is formulated as a difficult nonconvex MIP problem. In MMC studies, there are several convex relaxation works [16], [21], [41] on the label matrices $\mathbf{M} = \mathbf{y}\mathbf{y}^T$, where global minima can be achieved. We follow the convex relaxation method in [41] and construct a convex hull [42] on all feasible label vectors $\mathbf{y}$ directly, which is the tightest convex relaxation on $\mathbf{y}$.

According to the minimax theorem [83], the optimal objective of (9) is an upper bound of $\max_{\boldsymbol{\alpha}} \min_{\mathbf{y}} E(\mathbf{y};\boldsymbol{\alpha})$, which means that problem (9) can be rewritten as

$$\max_{\boldsymbol{\alpha}\in\mathcal{A}} \left\{ \max_{\theta} -\theta \quad \text{s.t. } \theta \geq -E(\mathbf{y}_k;\boldsymbol{\alpha}), \qquad \forall \mathbf{y}_k \in \mathcal{B}_0 \right\}. \tag{12}$$

We are to solve the subproblem in the brace of (12) in its dual, so that problem (12) can be formulated as the following concave–convex optimization problem:

$$\max_{\boldsymbol{\alpha}\in\mathcal{A}} \min_{\mathbf{y}'\in\mathcal{B}_1} E(\mathbf{y}';\boldsymbol{\alpha}) = \min_{\mathbf{y}'\in\mathcal{B}_1} \max_{\boldsymbol{\alpha}\in\mathcal{A}} E(\mathbf{y}';\boldsymbol{\alpha}) \tag{13}$$

where $\mathcal{B}_1 = \{\mathbf{y}' | \mathbf{y}' = \sum_{k:\mathbf{y}_k\in\mathcal{B}_0} \mu_k \mathbf{y}_k, \boldsymbol{\mu} \in \mathcal{M}\}$ is the convex hull of $\mathcal{B}_0$. Until now, we have constructed a convex hull directly on all feasible label vectors.

C. Solving the Convex Optimization Problem

Problem (13) [or (12)] can be also rewritten as

$$\min_{\boldsymbol{\mu}\in\mathcal{M}} \max_{\boldsymbol{\alpha}\in\mathcal{A}} E\left(\sum_{k:\mathbf{y}_k\in\mathcal{B}_0} \mu_k \mathbf{y}_k;\boldsymbol{\alpha} \right). \tag{14}$$

It is clear that the problem is convex on $\boldsymbol{\mu}$ and concave on $\boldsymbol{\alpha}$. However, solving problem (14) directly is difficult, since the lengths of unknown parameter vectors $\boldsymbol{\mu}$ and $\boldsymbol{\alpha}$ grow exponentially with the data set size.

In this subsection, we are to solve problem (14) approximately with the thoughts of CPA. An overview of the solution are first given as follows:

Layer 1 We first reduce the computational load on $\boldsymbol{\mu}$ by CPA. For each CPA iteration, we first solve the cutting-plane subproblem of the master problem (14), which will be shown as (15). Then, we calculate the most violated $\mathbf{y}$.

Layer 2 Because each cutting-plane subproblem (15) is also a concave–convex optimization problem, it can be approximately solved by the ELM algorithm (described in Section III-C1). However, because solving each problem (15) independently via ELM cannot guarantee the convergence of the CPA, we further propose to solve each cutting-plane subproblem (15) by ELM with all historical ELM information. The new method is called GELM.

Layer 3 As presented in Section III-C1, the ELM for problem (15) contains two steps. The first step is to get $\boldsymbol{\alpha}$ with fixed $\boldsymbol{\mu}$ by solving an SVR problem. The second step is to get $\boldsymbol{\mu}$ with fixed $\boldsymbol{\alpha}$ by the *projection* function. For the first step, because the parameter vector $\boldsymbol{\alpha}$ might be large, the CPSP algorithm, which is a combination of the CPA and the sparse-kernel estimation techniques, is employed to solve the SVR problem approximately. The new method is called sparse-kernel SVR (SKSVR).

Now, we present the aforementioned method in detail.

1) Layer 1: Solving Problem (14) via CPA: From the fact that most constraints of problem (12) can be inactive, we know that most elements of $\boldsymbol{\mu}$ in (14) can be zero, since $\boldsymbol{\mu}$ is the Lagrangian variable vector of the constraints in (12). Thus, we are to solve the MMC problem (14) iteratively by CPA [33]. The CPA iterates the following two steps until convergence.

The first step is to solve the current cutting-plane subproblem of (14), which is formulated as

$$\min_{\boldsymbol{\mu}\in\mathcal{M}_{|\Omega|}} \max_{\boldsymbol{\alpha}\in\mathcal{A}} E\left(\sum_{k=1}^{|\Omega|} \mu_k \mathbf{y}_k;\boldsymbol{\alpha} \right) \tag{15}$$

where the cutting-plane working constraint set $\boldsymbol{\Omega}$ is $\boldsymbol{\Omega} = \{\mathbf{y}_1, \ldots, \mathbf{y}_{|\Omega|}\}$, $|\Omega|$ is the size of $\boldsymbol{\Omega}$, and $\mathcal{M}_{|\Omega|} = \{\boldsymbol{\mu} | \sum_{k=1}^{|\Omega|} \mu_k = 1, \boldsymbol{\mu} \geq \mathbf{0}\}$. The method of solving (15) will be presented later in this subsection.

The second step is to calculate the most violated constraint $\mathbf{y}$ of problem (12).

Theorem 2: Given solution $(\boldsymbol{\mu},\boldsymbol{\alpha})$ of the cutting-plane subproblem (15), the most violated $\mathbf{y}$ can be obtained by solving the following simple binary-integer-linear-programming problem:

$$\min_{\mathbf{y}\in\mathcal{B}_0} \boldsymbol{\alpha}^T \mathbf{G}^T \mathbf{y} \tag{16}$$

which can be efficiently solved in the same way as [41, Proposition 2] in time $\mathcal{O}(n \log n)$ without any numeric optimization solver.

Proof: The proof of Theorem 2 is provided in Appendix B. ■

After obtaining the most violated $\mathbf{y}$, if $-E(\mathbf{y};\boldsymbol{\alpha}) \leq -E(\sum_{k=1}^{|\Omega|} \mu_k \mathbf{y}_k;\boldsymbol{\alpha}) + \eta$, the CPA can be stopped; otherwise, we add $\mathbf{y}$ to $\boldsymbol{\Omega}$ and continue, where η is a user-defined cutting-plane solution precision.

2) Layer 2: Solving the Cutting-Plane Subproblem (15) via GELM: At first glance, we can solve each cutting-plane subproblem (15) independently (locally) via ELM since the subproblem is a concave–convex optimization problem.

However, although the objective value of a reduced cutting-plane subproblem is monotonic w.r.t. the number of the ELM iterations, the ELM does not guarantee the convergence behavior of the master problem of the CPA.

To overcome the problem, we will try ELM to solve the master problem (14) globally but not to solve serial reduced problems (15) locally. The key point of solving the master problem (14) globally is to make sure that the objective value of the previous cutting-plane subproblem is an upper bound of that of the successive cutting-plane subproblem. For this, the ELM of the successive cutting-plane subproblem should start at the terminating point of the previous one by **inheriting all previous ELM solutions** that account for the objective value of the previous one.

Suppose the ith ELM solution of the (current) $|\Omega|$th cutting plane subproblem (15) is $(\boldsymbol{\mu}_{|\Omega|}^i, \boldsymbol{\alpha}_{|\Omega|}^i)$. Suppose the previous kth cutting-plane subproblem needs T_k ELM iterations to converge and yields T_k ELM solutions denoted as $\{(\boldsymbol{\mu}_k^i, \boldsymbol{\alpha}_k^i)\}_{i=1}^{T_k}$, where $k = 1, \ldots, |\Omega| - 1$.

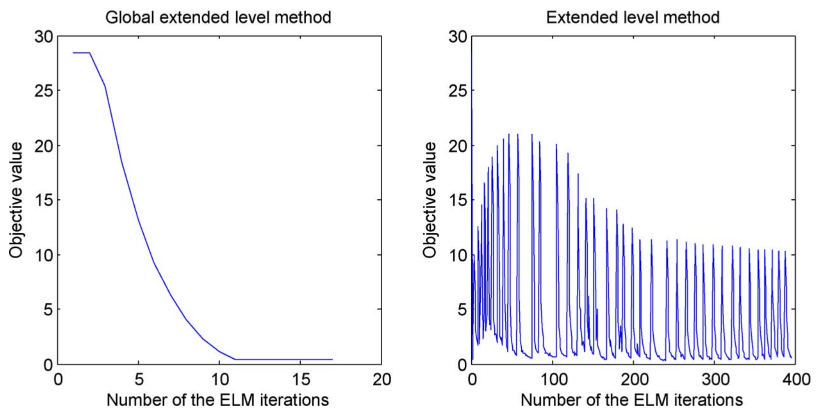

Fig. 1. Convergence behavior comparison of (left) the GELM- and (right) local ELM-based MMC.

Theorem 3: To guarantee the convergence of the CPA, the ELM solutions $\{(\boldsymbol{\mu}_k^i, \boldsymbol{\alpha}_k^i)\}_{i=1}^{T_k}$ of the kth cutting-plane subproblem (15) should be inherited by the $|\Omega|$th ($|\Omega| > k$) cutting-plane subproblem by adding each $\boldsymbol{\mu}_k^i$ with $|\Omega| - k$ zeros.

Proof: The proof of Theorem 3 is provided in Appendix C. ■

From this point of view, the CPA and the ELM algorithm are combined together. We call the combination of the CPA and the ELM as the GELM algorithm.

Fig. 1 gives a convergence behavior comparison of the GELM-based MMC (left) and the local ELM-based SVR-MMC (right). From the right figure, we can clearly see that, although the objective values of the ELM within any cutting-plane subproblem decrease w.r.t. the ELM iterations, the objective values of the cutting-plane subproblems do not rigorously decrease w.r.t. the number of the cutting-plane iterations, whereas we can see from the left figure that the GELM guarantees the convergence of the CPA.

The detailed processes of the GELM algorithm for a single cutting-plane subproblem (15) are presented below.

Initialization of the GELM: The first thing in this step is to inherit all historical ELM solutions for the (current) $|\Omega|$th cutting-plane subproblem (15). The inheritance can be realized by adding $\boldsymbol{\mu}_k^i$ with $|\Omega| - k$ zeros

$$\boldsymbol{\mu'}_k^i = \begin{bmatrix} \boldsymbol{\mu}_k^i \\ \mathbf{0}_{|\Omega|-k} \end{bmatrix} \quad \forall k = 1, \ldots, |\Omega| - 1, i = 1, \ldots, T_k \quad (17)$$

so as to make $\boldsymbol{\mu'}_k^i$ the same length as $\boldsymbol{\mu}$.

Main procedure of the GELM: After inheriting the historical ELM solutions, problem (15) can be solved in the same way as the ELM. It iterates two steps until the ELM converges.

a) Suppose we are currently at the uth ELM iteration of the $|\Omega|$th cutting-plane subproblem. We have all the last ELM solutions chronologically as

$$\{(\boldsymbol{\mu}^i, \boldsymbol{\alpha}^i)\}_{i=1}^{j-1} = \left\{ \left\{ \left(\boldsymbol{\mu'}_k^i, \boldsymbol{\alpha}_k^i \right) \right\}_{i=1}^{T_k} \right\}_{k=1}^{|\Omega|-1} \cup \left\{ \left(\boldsymbol{\mu}_{|\Omega|}^i, \boldsymbol{\alpha}_{|\Omega|}^i \right) \right\}_{i=1}^{u-1} \quad (18)$$

and solution $\boldsymbol{\mu}^j$ of current ELM iteration, where $j = \sum_{k=1}^{|\Omega|-1} T_k + u$ is the total number of all last ELM iterations.

In this step, we need to get $\boldsymbol{\alpha}^j$ by solving the following concave SVR problem:

$$\max_{\boldsymbol{\alpha} \in \mathcal{A}} E(\mathbf{y}^*; \boldsymbol{\alpha}) \quad (19)$$

with $\mathbf{y}^*$ defined as $\mathbf{y}^* = \sum_{k=1}^{|\Omega|} {\mu_k}^j \mathbf{y}_k$, where ${\mu_k}^j$ denotes the kth elements of $\boldsymbol{\mu}^j$.

Because problem (19) has 2^n unknown parameters, its difficult to solve it directly when the data set is large scale. Here, we employ the CPSP algorithm [34], [50], [51], [84] to solve problem (19) approximately with a sufficient estimation precision. The estimation algorithm is called SKSVR, which will be presented in Section IV-D. The SKSVR algorithm is denoted as the *SKSVR* function, which has linear time and storage complexities.

b) The lower and upper bounds of the master problem (15) is calculated as

$$\underline{E}^j = \min_{\boldsymbol{\mu}} h^j(\boldsymbol{\mu}); \overline{E}^j = \min_{1 \le i \le j} E(\mathbf{y}^*; \boldsymbol{\alpha}^i) \quad (20)$$

where $h^j(\boldsymbol{\mu}) = \max_{1 \le i \le j} E(\sum_{k=1}^{|\Omega|} \mu_k \mathbf{y}_k; \boldsymbol{\alpha}^i)$. Then, we use the bounds to construct the *level set* of ELM and obtain $\boldsymbol{\mu}^{j+1}$ by the *projection* function (defined in Section III-C1).

3) Layer 3: Solving the SVR Problem (19) via CPSP Algorithm: Because the CPSP-based SVR can be independently used as a supervised SVR toolbox, we present this part as a separate section. See Section IV-D.

Algorithm 1 SVR-MMC.

Input: Data set $\bar{\mathbf{x}} = \{\mathbf{x}_i\}_{i=1}^n$, regularization constant C, cutting plane solution precision η, ELM solution precision τ, CPSP solution precision ϵ, and parameter l that controls the class imbalance.

```
Initialization: Arbitrary initial constraint labels $\mathbf{y}_1$ ($-l \le \mathbf{1}^T\mathbf{y}_1 \le l$), cutting-plane working constraint set $\mathbf{\Omega} \leftarrow \{\mathbf{y}_1\}$, $|\Omega| \leftarrow 1$, $\mu_1 \leftarrow 1$, and the historical ELM solution set $\mathcal{J} \leftarrow \emptyset$, $j \leftarrow 0$.
Output: $\hat{\mathbf{y}}$.
1: repeat
2:   repeat
3:     $j \leftarrow j+1$
4:     $\mathbf{y}^* \leftarrow \sum_{k=1}^{|\Omega|} \mu_k^j \mathbf{y}_k$
5:     $(\boldsymbol{\alpha}^j, \mathbf{G}^j, \{\boldsymbol{\beta}^j, \hat{\mathbf{K}}^j\}) \leftarrow \mathrm{SKSVR}(\bar{\mathbf{x}}; \mathbf{y}^*; C; \epsilon)$
6:     $\mathcal{J} \leftarrow \mathcal{J} \cup (\boldsymbol{\mu}^j, \boldsymbol{\alpha}^j, \mathbf{G}^j, \{\boldsymbol{\beta}^j, \hat{\mathbf{K}}^j\})$
7:     $\mu^{j+1} \leftarrow projection(\mathcal{J}; \mathbf{\Omega})$
8:   until $\|\boldsymbol{\mu}^j - \boldsymbol{\mu}^{j+1}\|^2 \le \tau$
9:   $\mathbf{y}_{|\Omega|+1} \leftarrow \min_{\mathbf{y}\in\mathcal{B}_0} {\boldsymbol{\alpha}^j}^T {\mathbf{G}^j}^T \mathbf{y}$
10:  $\mathbf{\Omega} \leftarrow \mathbf{\Omega} \cup \mathbf{y}_{|\Omega|+1}$
11:  for $jj = 1, \ldots, j$ do
12:    $\boldsymbol{\mu}^{jj} \leftarrow [{\boldsymbol{\mu}^{jj}}^T, 0]^T$
13:  end for
14:  $\xi' \leftarrow -{\boldsymbol{\alpha}^j}^T {\mathbf{G}^j}^T \mathbf{y}_{|\Omega|+1}$, and $\xi \leftarrow -{\boldsymbol{\alpha}^j}^T {\mathbf{G}^j}^T \mathbf{y}^*$
15:  $|\Omega| \leftarrow |\Omega| + 1$
16: until $\xi' \le \xi + \eta$ or $\mathbf{\Omega}$ is unchanged
17: for $i = 1, \ldots, n$ do
18:   $\hat{y}_i \leftarrow sign(\sum_t \alpha_t^j \beta_t^j K(\mathbf{b}_t, \mathbf{x}_i))$
19: end for
```

D. Linear Time SKSVR

In this subsection, we will solve problem (19) in linear time and storage complexities in both the linear kernel scenario and the nonlinear kernel scenarios by CPSP [34], [50], [51]. Specifically, we will first solve problem (19) by CPA. Then, we will overcome the computational burden of nonlinear kernels via sparse-kernel estimation techniques [84]. Finally, the SKSVR algorithm is summarized in Algorithm 2.

Algorithm 2 *SKSVR.*

```
Input: Data set $\bar{\mathbf{x}} = \{\mathbf{x}_i\}_{i=1}^n$, label vector $\mathbf{y}^*$, regularization constant $C$, and CPSP solution precision $\epsilon$.
Initialization: Arbitrary initial CPSP constraint vector $\mathbf{g}_1$, CPSP working constraint set $\mathbf{\Omega}_g \leftarrow \{\mathbf{g}_1\}$, and $z \leftarrow 1$.
Output: $(\boldsymbol{\alpha}, \mathbf{G}_z, \{\boldsymbol{\beta}, \hat{\mathbf{K}}\})$.
1: repeat
2:   $(\beta_z, \mathbf{b}_z) \leftarrow \mathrm{estimate\_basis}(\bar{\mathbf{x}}, \mathbf{g}_z)$
3:   $\hat{\mathbf{K}} = [\langle\phi(\mathbf{b}_k), \phi(\mathbf{b}_l)\rangle]_{k,l=1}^z$
4:   Get $\mathbf{G}_z$ from $\mathbf{\Omega}_g$: $\mathbf{G}_z \leftarrow [\mathbf{g}_1, \ldots, \mathbf{g}_z]$
5:   Solve the quadratic programming problem (26), and get the objective value $O$ and $\boldsymbol{\alpha}$.
6:   $\hat{\mathbf{w}} \equiv \sum_{t=1}^z \alpha_t \beta_t \phi(\mathbf{b}_t)$
7:   Calculate the most violated $\mathbf{g}_{z+1}$ from (24).
8:   Renew $\mathbf{\Omega}_g$: $\mathbf{\Omega}_g \leftarrow \mathbf{\Omega}_g \cup \mathbf{g}_{z+1}$.
9:   Calculate $\hat{\xi}'$ from (25).
10:  $\hat{\xi} \leftarrow (O - (1/2)\|\hat{\mathbf{w}}\|^2)/C$
11:  $z \leftarrow z+1$
12: until $\hat{\xi}' \le \hat{\xi} + \epsilon$.
```

1) CPA: Solving the master problem (19) by CPA needs to iterates two steps.

The first step is to solve a cutting-plane subproblem, i.e.,

$$\max_{\boldsymbol{\alpha}\in\mathcal{A}_z} \boldsymbol{\alpha}^T \mathbf{G}_z^T \mathbf{y}^* - \frac{1}{2}\boldsymbol{\alpha}^T \mathbf{G}_z^T \mathbf{K} \mathbf{G}_z \boldsymbol{\alpha} \tag{21}$$

where the current cutting-plane working constraint set is $\mathbf{\Omega}_g = \{\mathbf{g}_1, \ldots, \mathbf{g}_z\}$ with z denoted as the size of $\mathbf{\Omega}_g$, $\mathcal{A}_z = \{\boldsymbol{\alpha}|\boldsymbol{\alpha}^T\mathbf{1} \le C, \alpha_t \ge 0, t = 1, \ldots, z\}$, and $\mathbf{G}_z = [\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_z]$. The primal of problem (21) is written as

$$\min_{\mathbf{w},\xi\ge 0} \frac{1}{2}\|\mathbf{w}\|^2 + C\xi \qquad \text{s.t. } \frac{1}{n}\sum_{i=1}^n g_{t,i} y_i^* - \frac{1}{n}\sum_{i=1}^n g_{t,i}\mathbf{w}^T\phi(\mathbf{x}_i) \le \xi, \quad \forall t = 1, \ldots, z \tag{22}$$

with $\mathbf{w}$ formulated as

$$\mathbf{w} = \sum_{t=1}^z \alpha_t \mathbf{\Psi}_t = \sum_{t=1}^z \alpha_t \left(\frac{1}{n}\sum_{i=1}^n g_{t,i}\phi(\mathbf{x}_i)\right) \tag{23}$$

and $\xi = (O - (1/2)\|\mathbf{w}\|^2)/C$, where O is the objective value of (21).

The second step is to calculate the most violated constraint $\mathbf{g}_{z+1}$.

Theorem 4: Given solution $(\mathbf{w}, \xi)$, the most violated constraint of problem (21) can be calculated as follows:

$$g_{z+1,i} = \begin{cases} 1, & \text{if } y_i^* - \mathbf{w}^T\phi(\mathbf{x}_i) \ge 0 \\ -1, & \text{otherwise.} \end{cases} \tag{24}$$

Proof: The most violated constraint $\mathbf{g}_{|\Omega|+1}$ is the one that would result in the largest ξ in (22). According to (56), the maximum value of ξ is calculated as

$$\xi' = \frac{1}{n}\sum_{i=1}^n \max_{g_{z+1,i}\in\{\pm 1\}} \left(g_{z+1,i}\left(y_i^* - \mathbf{w}^T\phi(\mathbf{x}_i)\right)\right). \tag{25}$$

The corresponding $\mathbf{g}_{z+1}$ of ξ' is calculated as (24). ■

If $\xi' \le \xi + \epsilon$, the CPA can be stopped; otherwise, we add $\mathbf{g}_{z+1}$ to $\mathbf{\Omega}_g$ and go to the next CPA iteration, where ϵ is the user-defined cutting-plane solution precision.

2) Sparse-Kernel Estimation: From (21)–(23), we can see that each constraint $\mathbf{g}_t$ relates to a computationally expensive operator $\mathbf{\Psi}_t = (1/n)\sum_i g_{t,i}\phi(\mathbf{x}_i)$, which causes a time complexity of $\mathcal{O}(zn)$ for $\mathbf{w}^T\phi(\mathbf{x})$ and $\mathcal{O}((zn)^2)$ for $\mathbf{G}_z^T\mathbf{K}\mathbf{G}_z$. In order to eliminate this expensive scaling behavior, we employ the basis vector estimation algorithm [38], [48], [84] to get an accurate sparse estimation of $\mathbf{\Psi}_t$ as $\hat{\mathbf{\Psi}}_t = \beta_t\phi(\mathbf{b}_t)$,[3] where $(\beta_t, \mathbf{b}_t)$ is called a basis vector in the *reduced set* of the kernel-induced feature space [48].

After the basis vector estimation, problem (21) is accurately approximated by the following problem:

$$\max_{\boldsymbol{\alpha}\in\mathcal{A}_z} \boldsymbol{\alpha}^T \mathbf{G}_z^T \mathbf{y}^* - \frac{1}{2}\boldsymbol{\alpha}^T \left(\hat{\mathbf{K}} \circ (\boldsymbol{\beta}\boldsymbol{\beta}^T)\right)\boldsymbol{\alpha} \tag{26}$$

[3]Only one basis vector is used for the estimation.

where operator $\circ$ denotes the element-wise product, $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_z]^T$, and $\hat{\mathbf{K}} = [\langle \phi(\mathbf{b}_k), \phi(\mathbf{b}_l)\rangle]_{k,l=1}^{z}$ is a sparse estimation of the original kernel $\mathbf{K}$.

Weight $\mathbf{w}$ is also accurately approximated by

$$\hat{\mathbf{w}} = \sum_{t=1}^{z} \alpha_t \beta_t \phi(\mathbf{b}_t). \quad (27)$$

With the basis vector estimation, the time complexity on $\mathbf{w}^T\phi(\mathbf{x})$ and $\mathbf{G}_z^T\mathbf{K}\mathbf{G}_z$ is lowered to $\mathcal{O}(z)$ and $\mathcal{O}(z^2)$, respectively, which are irrelevant to n. Moreover, solving problem (26) needs at most $\mathcal{O}(z^3)$ time. When the data set is large scale, $z \ll n$, which means that the computation of the SKSVR is mostly consumed on the basis vector estimations.

In this paper, three basis vector estimation algorithms are adopted.

- Linear kernel. The basis vector is calculated accurately as follows:

$$\beta_t = 1, \mathbf{b}_t = \frac{1}{n}\sum_{i=1}^{n} g_{t,i}\mathbf{x}_i \qquad \forall t = 1, \ldots, z. \quad (28)$$

  The time complexity for the basis vector estimation is $\mathcal{O}(sn)$, where s is the sparsity of the data set.
- Radial-basis-function (RBF) kernel. According to [84], the fixed point iteration approach presented in [38, Chapter 18] is used as the basis vector estimation algorithm. Its time complexity is $\mathcal{O}(rsn)$, where r is the average iteration number of the algorithm.
- General kernels. If the nonlinear kernels $\mathbf{K}$ have first-order derivatives, such as the polynomial kernel, a common approximation method for them was presented in [48]. It also has a linear time complexity $\mathcal{O}(rsn)$.

E. Overview of the Algorithm

First of all, what we want to solve is problem (14) [or its primal problem (12)], which is the tightest convex relaxation of the SVR-MMC objective (8). Because to solve problem (14) [or (12)] directly is difficult, we propose Algorithm 1 to solve it approximately with the *SKSVR* function described in Algorithm 2.

Algorithm 1 contains two loops, i.e., the outer cutting-plane loop and the inner ELM loop. Each outer cutting plane iteration constructs a new cutting-plane subproblem (15) by adding the most violated constraint (16) to the constraint set $\boldsymbol{\Omega}$. Because subproblem (15) is formulated as a concave–convex optimization problem, it is further solved by the inner ELM loop. One special point in Algorithm 1 is that all historical ELM information is saved in set $\mathcal{J}$, which combined the two loops as an integrate one.

Each inner ELM iteration contains two steps. The first step is to update $\boldsymbol{\alpha}$ by solving the concave optimization problem (18) with the *SKSVR* function. The second step is to update $\boldsymbol{\mu}$ by calling the *projection* function (defined in Section III-C1) with all inherited ELM solutions.

Particularly, because the first step of the inner ELM iteration aims to solve a computationally expensive SVR problem (18), we solve it approximately via CPSP algorithm, which is implemented as the *SKSVR* function (described in Section IV-D). The *SKSVR* function has linear time and storage complexities.

V. SVR-Based MKC

In this section, we will extend the single-kernel-based SVR-MMC algorithm to the multiple-kernel scenario.

A. Objective Formulation

The general form of the MKC objective can be generated from the combination of the original MMC objective (1) and the MKL objective (4), i.e.,

$$\min_{\mathbf{y}\in\mathcal{B}_0} \min_{\mathbf{v},b,\boldsymbol{\theta}\geq\mathbf{0}} \frac{1}{2}\sum_{q=1}^{Q}\frac{\|\mathbf{v}_q\|^2}{\theta_q} + C\ell\left(\{f_{\mathbf{v},b}(\mathbf{x}_i)\}_{i=1}^{n}, \mathbf{y}\right)$$
$$\text{s.t. } J(\boldsymbol{\theta}) \leq 1. \quad (29)$$

Taking multiple-kernel SVR (MKSVR) as the core of the MKC objective results in the following computationally difficult MIP problem:

$$\min_{\mathbf{y}\in\mathcal{B}_0}\left\{\min_{\mathbf{v},\boldsymbol{\theta}\in\Theta,\boldsymbol{\xi}\geq\mathbf{0},\boldsymbol{\xi}^*\geq\mathbf{0}} \frac{1}{2}\sum_{q=1}^{Q}\frac{\|\mathbf{v}_q\|^2}{\theta_q} + \frac{C}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*) \right.$$
$$\text{s.t. } y_i - \sum_{q=1}^{Q}\mathbf{v}_q^T\phi_q(\mathbf{x}_i) \leq \xi_i$$
$$\left. - y_i + \sum_{q=1}^{Q}\mathbf{v}_q^T\phi_q(\mathbf{x}_i) \leq \xi_i^* \qquad \forall i = 1, \ldots, n\right\}$$
$$(30)$$

where the optimization problem in the brackets is the objective of the Laplacian-loss MKSVR and the constraint $J(\boldsymbol{\theta}) \leq 1$ is specified as the L_1 norm regularization of $\boldsymbol{\theta}$, e.g., $\Theta = \{\boldsymbol{\theta} | \sum_{q=1}^{Q}\theta_q = 1, \boldsymbol{\theta} \geq \mathbf{0}\}$. Note that there are several discussions on the constraint of $\boldsymbol{\theta}$, and the proposed method is not limited to the L_1 norm.

After n-slack to 1-slack reduction, objective (30) can be reformulated as

$$\min_{\mathbf{y}\in\mathcal{B}_0}\min_{\boldsymbol{\theta}\in\Theta}\left\{\min_{\mathbf{v},\xi\geq0}\frac{1}{2}\sum_{q=1}^{Q}\frac{\|\mathbf{v}_q\|^2}{\theta_q} + C\xi \right.$$
$$\text{s.t. } \frac{1}{n}\sum_{i=1}^{n}g_iy_i - \frac{1}{n}\sum_{i=1}^{n}g_i\sum_{q=1}^{Q}\mathbf{v}_q^T\phi_q(\mathbf{x}_i) \leq \xi$$
$$\left. \forall \mathbf{g} \in \{1, -1\}^n\right\}. \quad (31)$$

Problems (30) and (31) are equivalent in the following theorem.

Theorem 5: Any solution $(\mathbf{v}, \mathbf{y}, \boldsymbol{\theta})$ of problem (31) is also a solution of problem (30) and *vice versa*, with $\xi = (1/n)\sum_{i=1}^{n}(\xi_i + \xi_i^*)$.

Proof: The proof is similar with the proof of Theorem 1. ■

As a kernel-based method, it is more convenient to solve problem (31) in the dual of the MKSVR problem. Doing so will derive the following problem:

$$\min_{\mathbf{y}\in\mathcal{B}_0}\min_{\boldsymbol{\theta}\in\Theta}\max_{\boldsymbol{\alpha}\in\mathcal{A}} E(\mathbf{y};\boldsymbol{\theta};\boldsymbol{\alpha}) \tag{32}$$

where $E(\mathbf{y};\boldsymbol{\theta};\boldsymbol{\alpha})$ is defined as

$$E(\mathbf{y};\boldsymbol{\theta};\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T\mathbf{G}^T\mathbf{y} - \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{G}^T\left(\sum_{q=1}^{Q}\theta_q\mathbf{K}_q\right)\mathbf{G}\boldsymbol{\alpha} \tag{33}$$

with $\mathbf{K}_q = [K_q(\mathbf{x}_i,\mathbf{x}_j)]_{i,j=1}^n$ denoted as the qth base kernel matrix. We can also get $\mathbf{v}_q$ as

$$\mathbf{v}_q = \theta_q\sum_{k=1}^{2^n}\alpha_k\left(\frac{1}{n}\sum_{i=1}^{n}g_{k,i}\phi_q(\mathbf{x}_i)\right), \quad q=1,\ldots,Q. \tag{34}$$

Algorithm 3 SVR-MKC.

Input: Data set $\bar{\mathbf{x}} = \{\mathbf{x}_i\}_{i=1}^n$, regularization constant C, cutting plane solution precision η, ELM solution precision τ, CPSP solution precision ϵ, and parameter l that controls the class imbalance.
Initialization: Initial kernel weights $\theta_q = 1/Q$, $q=1,\ldots,Q$, arbitrary initial constraint labels $\mathbf{y}_1$ $(-l \le \mathbf{1}^T\mathbf{y}_1 \le l)$, cutting-plane working constraint set $\boldsymbol{\Omega} \leftarrow \{\mathbf{y}_1\}$, $|\Omega| \leftarrow 1$, $\mu_1 \leftarrow 1$, and the historical ELM solution set $\mathcal{J} \leftarrow \emptyset$, $j \leftarrow 0$.
Output: $\hat{\mathbf{y}}$.

```
1: repeat
2:   repeat
3:     $j \leftarrow j+1$
4:     $\mathbf{y}^* \leftarrow \sum_{k=1}^{|\Omega|}\mu_k^j\mathbf{y}_k$
5:     $(\boldsymbol{\alpha}^j, \mathbf{G}^j, \{\boldsymbol{\beta}_q^j, \hat{\mathbf{K}}_q^j\}_{q=1}^Q) \leftarrow$
           $\mathrm{MKSVR}(\bar{\mathbf{x}};\mathbf{y}^*;\boldsymbol{\theta}^j;C;\epsilon)$
6:     $\mathcal{J} \leftarrow \mathcal{J} \cup (\boldsymbol{\mu}^j, \boldsymbol{\theta}^j, \boldsymbol{\alpha}^j, \mathbf{G}^j, \{\boldsymbol{\beta}_q^j, \hat{\mathbf{K}}_q^j\}_{q=1}^Q)$
7:     $(\boldsymbol{\mu}^{j+1}, \boldsymbol{\theta}^{j+1}) \leftarrow projection(\mathcal{J};\boldsymbol{\Omega})$
8:   until
         $\left\|\begin{bmatrix}\boldsymbol{\mu}^j\\ \boldsymbol{\theta}^j\end{bmatrix} - \begin{bmatrix}\boldsymbol{\mu}^{j+1}\\ \boldsymbol{\theta}^{j+1}\end{bmatrix}\right\|^2 \le \tau$
9:   $\mathbf{y}_{|\Omega|+1} \leftarrow \min_{\mathbf{y}\in\mathcal{B}_0}{\boldsymbol{\alpha}^j}^T{\mathbf{G}^j}^T\mathbf{y}$
10:  $\boldsymbol{\Omega} \leftarrow \boldsymbol{\Omega} \cup \mathbf{y}_{|\Omega|+1}$
11:  for $jj = 1,\ldots,j$ do
12:    $\boldsymbol{\mu}^{jj} \leftarrow [{\boldsymbol{\mu}^{jj}}^T, 0]^T$
13:  end for
14:  $\xi' \leftarrow -{\boldsymbol{\alpha}^j}^T{\mathbf{G}^j}^T\mathbf{y}_{|\Omega|+1}$, and $\xi \leftarrow -{\boldsymbol{\alpha}^j}^T{\mathbf{G}^j}^T\mathbf{y}^*$
15:  $|\Omega| \leftarrow |\Omega|+1$
16: until $\xi' \le \xi + \eta$ or $\boldsymbol{\Omega}$ is unchanged
17: for $i = 1,\ldots,n$ do
18:   $\hat{y}_i \leftarrow sign(\sum_{q=1}^Q\theta_q^j\sum_t\alpha_t^j\beta_{q,t}^jK_q(\mathbf{b}_{q,t},\mathbf{x}_i))$
19: end for
```

Algorithm 4 *MKSVR.*

Input: Data set $\bar{\mathbf{x}} = \{\mathbf{x}_i\}_{i=1}^n$, label vector $\mathbf{y}^*$, kernel weights $\boldsymbol{\theta}$, regularization constant C, and CPSP solution precision ϵ.
Initialization: Arbitrary initial CPSP constraint vector $\mathbf{g}_1$, CPSP working constraint set $\boldsymbol{\Omega}_g \leftarrow \{\mathbf{g}_1\}$, and $z \leftarrow 1$.
Output:$(\boldsymbol{\alpha}, \mathbf{G}_z, \{\boldsymbol{\beta}_q, \hat{\mathbf{K}}_q\}_{q=1}^Q)$.

```
1: repeat
2:   for $q = 1,\ldots,Q$ do
3:     $(\beta_{q,z}, \mathbf{b}_{q,z}) \leftarrow \mathrm{estimate\_basis}^{(q)}(\bar{\mathbf{x}}, \mathbf{g}_z)$
4:     $\hat{\mathbf{K}}_q = [\langle\phi_q(\mathbf{b}_{q,k}), \phi_q(\mathbf{b}_{q,l})\rangle]_{k,l=1}^z$
5:   end for
6:   Get $\mathbf{G}_z$ from $\boldsymbol{\Omega}_g$: $\mathbf{G}_z \leftarrow [\mathbf{g}_1,\ldots,\mathbf{g}_z]$
7:   Solve the following quadratic programming problem,
     and get the objective value $O$ and $\boldsymbol{\alpha}$:
       $$\max_{\boldsymbol{\alpha}\in\mathcal{A}_z}\boldsymbol{\alpha}^T\mathbf{G}_z^T\mathbf{y}^* - \frac{1}{2}\boldsymbol{\alpha}^T\left(\sum_{q=1}^{Q}\theta_q\left(\hat{\mathbf{K}}_q\circ(\boldsymbol{\beta}_q\boldsymbol{\beta}_q^T)\right)\right)\boldsymbol{\alpha}$$
     where operator $\circ$ denotes the element-wise product and
       $\mathcal{A}_z = \{\boldsymbol{\alpha}|\boldsymbol{\alpha}^T\mathbf{1} \le C, \alpha_t \ge 0, t=1,\ldots,z\}$.
8:   $\hat{\mathbf{v}}_q \equiv \theta_q\sum_{t=1}^z\alpha_t\beta_{q,t}\phi_q(\mathbf{b}_{q,t})$, $q=1,\ldots,Q$
9:   Calculate the most violated $\mathbf{g}_{z+1}$ from
       $$g_{z+1,i} = \begin{cases}1, & \text{if } y_i^* - \sum_{q=1}^Q\hat{\mathbf{v}}_q\phi_q(\mathbf{x}_i) \ge 0\\ -1, & \text{otherwise.}\end{cases}$$
10:  Renew $\boldsymbol{\Omega}_g$: $\boldsymbol{\Omega}_g \leftarrow \boldsymbol{\Omega}_g \cup \mathbf{g}_{z+1}$.
11:  Calculate $\hat{\xi}'$ by
       $$\hat{\xi}' = \frac{1}{n}\sum_{i=1}^n\max_{g_{z+1,i}\in\{\pm1\}}\left(g_{z+1,i}\left(y_i^* - \sum_{q=1}^Q\hat{\mathbf{v}}_q^T\phi_q(\mathbf{x}_i)\right)\right)$$
12:  $\hat{\xi} \leftarrow (O - (1/2)\sum_{q=1}^Q\|\hat{\mathbf{v}}_q\|^2/\theta_q)/C$
13:  $z \leftarrow z+1$
14: until $\hat{\xi}' \le \hat{\xi} + \epsilon$.
```

B. Convex Relaxation

As the SVR-MMC did in Section IV-B, problem (32) can be rewritten

$$\max_{\boldsymbol{\alpha}\in\mathcal{A}}\min_{\boldsymbol{\theta}}\left\{\max_u -u \quad \text{s.t. } u \ge -E(\mathbf{y}_k;\boldsymbol{\theta};\boldsymbol{\alpha}), \forall k: \mathbf{y}_k\in\mathcal{B}_0\right\}. \tag{35}$$

Its dual is formulated as the following optimization problem:

$$\max_{\boldsymbol{\alpha}\in\mathcal{A}}\min_{\boldsymbol{\theta}\in\Theta}\min_{\mathbf{y}'\in\mathcal{B}_1}E(\mathbf{y}';\boldsymbol{\theta};\boldsymbol{\alpha}) = \min_{\mathbf{y}'\in\mathcal{B}_1}\min_{\boldsymbol{\theta}\in\Theta}\max_{\boldsymbol{\alpha}\in\mathcal{A}}E(\mathbf{y}';\boldsymbol{\theta};\boldsymbol{\alpha}) \tag{36}$$

where $\mathcal{B}_1 = \{\mathbf{y}'|\mathbf{y}' = \sum_{k:\mathbf{y}_k\in\mathcal{B}_0}\mu_k\mathbf{y}_k, \boldsymbol{\mu}\in\mathcal{M}\}$ is the convex hull of $\mathcal{B}_0$.

C. Solving the Convex Optimization Problem

The convex optimization problem (36) can be solved in a similar way with the algorithm presented in Section IV-C.

In this subsection, we emphasize the difference between the solution used in SVR-MMC and the solution used here.

1) Layer 1: Solving Problem (36) via CPA: Problem (36) is solved by CPA. The CPA iterates the following two steps until convergence.

The first step is to solve the following cutting-plane subproblem:

$$\min_{\boldsymbol{\mu}\in\mathcal{M}_{|\Omega|},\boldsymbol{\theta}\in\Theta}\max_{\boldsymbol{\alpha}\in\mathcal{A}} E\left(\sum_{k=1}^{|\Omega|}\mu_k\mathbf{y}_k;\boldsymbol{\theta};\boldsymbol{\alpha}\right). \tag{37}$$

The second step is to calculate the most violated $\mathbf{y}$ of problem (37) in the same way as Theorem 2.

2) Layer 2: Solving the Cutting-Plane Subproblem (37) via GELM: We also solve problem (37) by GELM described in Section IV-C2. Some differences are emphasized below.

Initialization of the GELM: We inherit not only the historical $\boldsymbol{\mu}$ but also the historical $\boldsymbol{\theta}$ from previous CPA iterations.

Main procedure of the GELM: After inheriting the historical ELM solutions, problem (37) is solved by iterating the following two steps until the ELM converges.

a) Given $\boldsymbol{\mu}$ and $\boldsymbol{\theta}$, we solve an MKSVR problem but not a single-kernel SVR problem, i.e.,

$$\max_{\boldsymbol{\alpha}\in\mathcal{A}} E\left(\sum_{k=1}^{|\Omega|}\mu_k\mathbf{y}_k;\boldsymbol{\theta};\boldsymbol{\alpha}\right). \tag{38}$$

Its solution will be presented in ***Layer 3*** of this subsection.

b) Given $\boldsymbol{\alpha}$, we update $\boldsymbol{\mu}$ and $\boldsymbol{\theta}$ simultaneously by the *projection* function presented in Section III-C1.

3) Layer 3: Solving the MKSVR Problem (38) via CPSP Algorithm: The sparse-kernel MKSVR algorithm is summarized in Algorithm 4. For simplicity of this paper, we will not present Algorithm 4 in detail. The most significant difference between Algorithms 2 (*SKSVR* function) and 4 is that, in Algorithm 4, we estimate a basis vector for each base kernel K_q in a single cutting-plane iteration.

D. Overview of the Algorithm

The SVR-MKC algorithm is a weighted version of the SVR-MMC algorithm, which is summarized in Algorithm 3. There are one significant difference between SVR-MMC (Algorithm 1) and SVR-MKC (Algorithm 3). That is, SVR-MMC solves an *SKSVR*} problem in a single ELM iteration, whereas SVR-MKC solves an *MKSVR*} problem in a single ELM iteration. However, both *SKSVR*} and *MKSVR*} have similar time and storage complexities, so as SVR-MMC and SVR-MKC.

Although SVR-MKC and SVR-MMC are very similar, as shown in the experimental section, the SVR-MKC algorithm can achieve more robust performance than the SVR-MMC algorithm by automatically selecting the most suitable combination of the base kernels.

VI. SVR-Based M3C

In this section, we will extend the binary-class SVR-MMC algorithm to the multiclass scenario.

A. Objective Formulation

Unlike the uniform objectives of the MMC and the MKC [see (1) and (29), respectively], we cannot get a uniform objective for M3C. Because of this, we will give out our special M3C objective "suddenly."

Given a P class clustering problem, the sample $\mathbf{x}$'s label y belongs to the integer set $\{1,2,\ldots,P\}$. Now, we extend label y to a P-dimensional **row** vector [77], [78], [80], denoted by $\bar{\mathbf{y}}$. Suppose $y=k$, $k\in\{1,\ldots,P\}$, the label vector $\bar{\mathbf{y}}$ takes 1 for the kth element and $-1/(P-1)$ for the others. For instance, if the ith sample falls into the first class, then $\bar{\mathbf{y}}_i=[1,-1/(P-1),\ldots,-1/(P-1)]$. Here, we define set $\mathcal{B}_{\bar{\mathbf{y}}}$ for all possible $\bar{\mathbf{y}}$, i.e.,

$$\mathcal{B}_{\bar{\mathbf{y}}}=\left\{\bar{\mathbf{y}}\,\middle|\,\left\{\bar{y}_p=\begin{cases}1, & \text{if } p=y\\ -\frac{1}{P-1}, & \text{otherwise.}\end{cases}\quad\forall p=1,\ldots,P\right\}\right.$$
$$\left.\forall y=1,\ldots,P\right\}$$

We propose the following SVR-based M3C objective:

$$\min_{\mathbf{Y}\in\mathcal{B}_2}\left\{\min_{\mathbf{W}}\frac{1}{2}\sum_{p=1}^{P}\|\mathbf{w}_p\|^2+\frac{C}{n}\sum_{p=1}^{P}\sum_{i=1}^{n}\left(\xi_{i,p}+\xi_{i,p}^*\right)\right.$$
$$\text{s.t. } \bar{y}_{i,p}-\mathbf{w}_p^T\phi(\mathbf{x}_i)\le\xi_{i,p},\ -\bar{y}_{i,p}+\mathbf{w}_p^T\phi(\mathbf{x}_i)\le\xi_{i,p}^*,$$
$$\left.\xi_{i,p}\ge 0,\ \xi_{i,p}^*\ge 0\qquad\forall p=1,\ldots,P,\ \forall i=1,\ldots,n\right\} \tag{39}$$

where the optimization problem in the brackets is the objective of the "multiclass SVR," $\mathbf{W}=[\mathbf{w}_1,\ldots,\mathbf{w}_P]$, the unknown label $\mathbf{Y}=[\bar{\mathbf{y}}_1^T,\ldots,\bar{\mathbf{y}}_n^T]^T$ is an $n\times P$ matrix, and set $\mathcal{B}_2$ is defined as

$$\mathcal{B}_2\triangleq\left\{\mathbf{Y}\,\middle|\,\begin{cases}-\frac{l_p}{P-1}\le\bar{\bar{\mathbf{y}}}_p^T\mathbf{1}\le l_p, & \forall p=1,\ldots,P,\\ \bar{\mathbf{y}}_i\in\mathcal{B}_{\bar{\mathbf{y}}}, & \forall i=1,\ldots,n.\end{cases}\right\}$$

where $\bar{\bar{\mathbf{y}}}_p=[\bar{y}_{1,p},\ldots,\bar{y}_{n,p}]^T$ denotes the pth column of $\mathbf{Y}$ and $\{l_p\}_{p=1}^P$ are user-defined constants for controlling class balance. Oftentimes, we set $l_1=l_2=,\ldots,=l_P=l$ for simplicity. Hence, set $\mathcal{B}_2$ becomes

$$\mathcal{B}_2\triangleq\left\{\mathbf{Y}\,\middle|\,\begin{cases}-\frac{l}{P-1}\le\bar{\bar{\mathbf{y}}}_p^T\mathbf{1}\le l & \forall p=1,\ldots,P,\\ \bar{\mathbf{y}}_i\in\mathcal{B}_{\bar{\mathbf{y}}} & \forall i=1,\ldots,n.\end{cases}\right\}. \tag{40}$$

Accordingly, an P-tuple of separating functions $\mathbf{f}(\mathbf{x})=\{f_1(\mathbf{x}),\ldots,f_P(\mathbf{x})\}$ is defined, where $f_p(\mathbf{x})=\mathbf{w}_p^T\phi(\mathbf{x})$. Once the optimal $\mathbf{W}$ is obtained, $\mathbf{x}$ is predicted as

$$y=\arg\max_p f_p(\mathbf{x})=\arg\max_p\mathbf{w}_p^T\phi(\mathbf{x}). \tag{41}$$

Now, we do an n-slack to 1-slack reduction for each class of objective (39) separately, which results in the following equivalent problem:

$$\min_{\mathbf{Y}\in\mathcal{B}_2}\left\{\min_{\mathbf{w},\boldsymbol{\xi}\geq\mathbf{0}}\frac{1}{2}\sum_{p=1}^{P}\|\mathbf{w}_p\|^2+C\sum_{p=1}^{P}\xi_p\right.$$
$$\text{s.t.}\ \frac{1}{n}\sum_{i=1}^{n}g_{i,p}\bar{y}_{i,p}-\frac{1}{n}\sum_{i=1}^{n}g_{i,p}\sum_{p=1}^{P}\mathbf{w}_p^T\phi(\mathbf{x}_i)\leq\xi_p$$
$$\left.\forall\mathbf{g}_p\in\{1,-1\}^n\quad\forall p=,\ldots,P\right\}\tag{42}$$

where $\boldsymbol{\xi}=[\xi_1,\ldots,\xi_P]^T$. The equivalence of problems (39) and (42) is supported by the following theorem.

Theorem 6: Any solution $(\mathbf{W},\mathbf{Y})$ of problem (39) is also a solution of problem (42) and *vice versa*, with $\sum_{p=1}^{P}\xi_p=(1/n)\sum_{p=1}^{P}\sum_{i=1}^{n}(\xi_{i,p}+\xi_{i,p}^*)$.

Proof: The proof is similar with the proof of Theorem 1. ■

As we did in SVR-MMC and SVR-MKC, we also solve SVR-M3C in the dual of the problem in the brackets of (42), which results in the following problem:

$$\min_{\mathbf{Y}\in\mathcal{B}_2}\max_{\boldsymbol{\alpha}\in\mathcal{A}^P}E_\Sigma(\mathbf{Y};\alpha)\triangleq\min_{\mathbf{Y}\in\mathcal{B}_2}\max_{\{\boldsymbol{\alpha}_p\in\mathcal{A}\}_{p=1}^P}\sum_{p=1}^{P}E_p(\bar{\bar{\mathbf{y}}}_p;\alpha_p)\tag{43}$$

where $\boldsymbol{\alpha}_p=\{\alpha_{i,p}\}_{i=1}^{2^n}$ is the dual variable vector of the pth class, $\boldsymbol{\alpha}=[\boldsymbol{\alpha}_1^T,\ldots,\boldsymbol{\alpha}_P^T]^T$, $\mathcal{A}^P=\mathcal{A}\times,\ldots,\times\mathcal{A}$, $E_\Sigma(\mathbf{Y};\alpha)\triangleq\sum_{p=1}^{P}E_p(\bar{\bar{\mathbf{y}}}_p;\alpha_p)$, and $E_p(\bar{\bar{\mathbf{y}}}_p;\alpha_p)$ is defined as

$$E_p(\bar{\bar{\mathbf{y}}}_p;\boldsymbol{\alpha}_p)=\boldsymbol{\alpha}_p^T\mathbf{G}_p^T\bar{\bar{\mathbf{y}}}_p-\frac{1}{2}\boldsymbol{\alpha}_p^T\mathbf{G}_p^T\mathbf{K}\mathbf{G}_p\boldsymbol{\alpha}_p.\tag{44}$$

We can also get $\mathbf{W}$ as

$$\mathbf{w}_p=\sum_{k=1}^{2^n}\alpha_{k,p}\left(\frac{1}{n}\sum_{i=1}^{n}g_{k,i,p}\phi(\mathbf{x}_i)\right),\quad p=1,\ldots,P.\tag{45}$$

B. Convex Relaxation

As the SVR-MMC did in Section IV-B, problem (43) can be written as

$$\max_{\boldsymbol{\alpha}\in\mathcal{A}^P}\left\{\max_{\theta}-\theta\quad\text{s.t.}\ \theta\geq-E_\Sigma(\mathbf{Y}_k;\alpha)\quad\forall\mathbf{Y}_k\in\mathcal{B}_2\right\}.\tag{46}$$

Solving the subproblem in the brace of (46) in its dual, we can reformulate problem (46) as the following problem:

$$\max_{\boldsymbol{\alpha}\in\mathcal{A}^P}\min_{\mathbf{Y}'\in\mathcal{B}_3}E_\Sigma(\mathbf{Y}';\boldsymbol{\alpha})=\min_{\mathbf{Y}'\in\mathcal{B}_3}\max_{\boldsymbol{\alpha}\in\mathcal{A}^P}E_\Sigma(\mathbf{Y}';\boldsymbol{\alpha})\tag{47}$$

where $\mathcal{B}_3=\{\mathbf{Y}'|\mathbf{Y}'=\sum_{k:\mathbf{Y}_k\in\mathcal{B}_2}\mu_k\mathbf{Y}_k,\boldsymbol{\mu}\in\mathcal{M}\}$ is the convex hull of $\mathcal{B}_2$. Until now, we have constructed a convex hull on all feasible label matrices of the M3C problem.

C. Solving the Convex Optimization Problem

The convex optimization problem (47) is solved in a similar way with the algorithm presented in Section IV-C.

1) Layer 1: Solving Problem (47) via CPA: Like the SVR-MMC, the CPA is utilized to solve the convex optimization problem (47), which iterates the following two steps.

The first step is to solve the following cutting-plane subproblem via ELM with all historical ELM solutions:

$$\min_{\boldsymbol{\mu}\in\mathcal{M}_{|\Omega|}}\max_{\boldsymbol{\alpha}\in\mathcal{A}^P}E_\Sigma\left(\sum_{k=1}^{|\Omega|}\mu_k\mathbf{Y}_k;\boldsymbol{\alpha}\right).\tag{48}$$

The second step is to calculate the most violated $\mathbf{Y}$ of the CPA.

Theorem 7: Given solution $(\boldsymbol{\mu},\boldsymbol{\alpha})$ of the cutting-plane subproblem (48), the most violated $\mathbf{Y}$ can be obtained by solving the following problem:

$$\min_{\mathbf{Y}\in\mathcal{B}_2}\sum_{p=1}^{P}\boldsymbol{\alpha}_p^T\mathbf{G}_p^T\bar{\bar{\mathbf{y}}}_p.\tag{49}$$

Proof: The proof of Theorem 7 is provided in Appendix D. ■

We further propose Algorithm 6 to solve problem (49). The correctness of Algorithm 6 is supported by Theorem 8.

Theorem 8: Output $\mathbf{Y}$ of Algorithm 6 is the solution of problem (49), which can be obtained in time $\mathcal{O}(Pn\log Pn)$.

Proof: The proof of Theorem 8 is provided in Appendix E. ■

2) Layer 2: Solving the Cutting-Plane Subproblem (48) via GELM: The GELM first inherits historical ELM solutions as SVR-MMC did in Section IV-C2 and then iterates the following two steps until convergence.

a) Given $\boldsymbol{\mu}$, we solve the "multiclass" SVR problem, i.e.,

$$\max_{\boldsymbol{\alpha}\in\mathcal{A}^P}E_\Sigma\left(\sum_{k=1}^{|\Omega|}\mu_k\mathbf{Y}_k;\boldsymbol{\alpha}\right).\tag{50}$$

Its solution will be presented in ***Layer 3*** of this subsection.

b) Given $\boldsymbol{\alpha}$, we update $\boldsymbol{\mu}$ by the *projection* function presented in Section III-C1.

3) Layer 3: Solving the Multiclass SVR Problem (51) via CPSP Algorithm: Because μ_k (so as to $\mathbf{Y}_k$) of problem (50) is known, problem (50) is in fact a sum of P-independent SVR problems, i.e.,

$$\sum_{p=1}^{P}\max_{\boldsymbol{\alpha}_p\in\mathcal{A}}E_p\left(\sum_{k=1}^{|\Omega|}\mu_k\bar{\bar{\mathbf{y}}}_{k,p};\boldsymbol{\alpha}_p\right).\tag{51}$$

We solve each SVR problem in (51) independently by the *SKSVR* function (Algorithm 2) and combine their solutions.

D. Overview of the Algorithm

The SVR-M3C algorithm is a multiclass extension of the SVR-MMC algorithm. It is summarized in Algorithm 5.

There are two significant differences between SVR-MMC (Algorithm 1) and SVR-M3C (Algorithm 5). The first difference is that SVR-MMC gets the most violated label vector $\mathbf{y}$ from set $\mathcal{B}_0$, whereas SVR-M3C gets the most violated label matrix $\mathbf{Y}$ from set $\mathcal{B}_2$ that are more complicated than $\mathcal{B}_0$. The second difference is that SVR-MMC solves just one *SKSVR* problem in a single ELM iteration, whereas SVR-M3C solves one *SKSVR* problem per class in a single ELM iteration. However, both of the algorithms have the same order of time and storage complexities.

Algorithm 5 SVR-M3C.

Input: Data set $\bar{\mathbf{x}} = \{\mathbf{x}_i\}_{i=1}^n$, regularization constant C, cutting-plane solution precision η, ELM solution precision τ, CPSP solution precision ϵ, parameter l that controls the class imbalance, and the number of clusters P
Initialization: Arbitrary initial constraint label matrix $\mathbf{Y}_1$ ($\mathbf{Y}_1 \in \mathcal{B}_2$), cutting-plane working constraint set $\mathbf{\Omega} \leftarrow \{\mathbf{Y}_1\}$, $|\Omega| \leftarrow 1$, $\mu_1 \leftarrow 1$, and the historical ELM solution set $\mathcal{J} \leftarrow \emptyset$, $j \leftarrow 0$.
Output: $\hat{\mathbf{y}}$.

```
1: repeat
2:   repeat
3:     $j \leftarrow j+1$
4:     $\mathbf{Y}^* \leftarrow \sum_{k=1}^{|\Omega|} \mu_k^j \mathbf{Y}_k$
5:     for $p = 1, \ldots, P$ do
6:       $(\boldsymbol{\alpha}_p^j, \mathbf{G}_p^j, \boldsymbol{\beta}_p^j, \hat{\mathbf{K}}_p^j) \leftarrow \mathrm{SKSVR}(\bar{\mathbf{x}}; \bar{\bar{\mathbf{y}}}_p^*; C; \epsilon)$
7:     end for
8:     $\mathcal{J} \leftarrow \mathcal{J} \cup (\boldsymbol{\mu}^j, \{\boldsymbol{\alpha}_p^j, \mathbf{G}_p^j, \boldsymbol{\beta}_p^j, \hat{\mathbf{K}}_p^j\}_{p=1}^P)$
9:     $(\boldsymbol{\mu}^{j+1}, \boldsymbol{\theta}^{j+1}) \leftarrow projection(\mathcal{J}; \mathbf{\Omega})$
10:  until $\|\boldsymbol{\mu}^j - \boldsymbol{\mu}^{j+1}\|^2 \le \tau$
11:  Solving the following optimization problem by Algorithm 6:
     $\mathbf{Y}_{|\Omega|+1} \leftarrow \min_{\mathbf{Y}\in\mathcal{B}_2} \sum_{p=1}^P {\boldsymbol{\alpha}_p^j}^T {\mathbf{G}_p^j}^T \bar{\bar{\mathbf{y}}}_p$
12:  $\mathbf{\Omega} \leftarrow \mathbf{\Omega} \cup \mathbf{Y}_{|\Omega|+1}$
13:  for $jj = 1, \ldots, j$ do
14:    $\boldsymbol{\mu}^{jj} \leftarrow [{\boldsymbol{\mu}^{jj}}^T, 0]^T$
15:  end for
16:  for $p = 1, \ldots, P$ do
17:    $\xi'_p \leftarrow -\sum_{p=1}^P {\boldsymbol{\alpha}_p^j}^T {\mathbf{G}_p^j}^T \bar{\bar{\mathbf{y}}}_{|\Omega|+1,p}$, and
       $\xi_p \leftarrow -\sum_{p=1}^P {\boldsymbol{\alpha}_p^j}^T {\mathbf{G}_p^j}^T \bar{\bar{\mathbf{y}}}_p^*$
18:  end for
19:  $|\Omega| \leftarrow |\Omega| + 1$
20: until $\sum_{p=1}^P \xi'_p \le \sum_{p=1}^P \xi_p + \eta$ or $\mathbf{\Omega}$ is unchanged
21: for $i = 1, \ldots, n$ do
22:   $\hat{y}_i \leftarrow \arg\max_p (\sum_t \alpha_{t,p}^j \beta_{t,p}^j K_p(\mathbf{b}_{t,p}, \mathbf{x}_i))$
23: end for
```

Algorithm 6 Calculating the most violated $\mathbf{Y}$.

Input: $\{\boldsymbol{\alpha}_p^T \mathbf{G}_p^T\}_{p=1}^P$ and constant l for the class balance.
Initialization: Let $\mathbf{c} = [\boldsymbol{\alpha}_1^T \mathbf{G}_1^T, \ldots, \boldsymbol{\alpha}_P^T \mathbf{G}_P^T]^T$, set the initial values of all elements of $\mathbf{Y}$ to $-1/(P-1)$, let $\mathbf{z} = [\bar{\bar{\mathbf{y}}}_1^T, \ldots, \bar{\bar{\mathbf{y}}}_P^T]^T$, and let the mask vector $\mathbf{m} = \mathbf{0}_{(nP)\times 1}$, initial objective value $O \leftarrow -nP/(P-1)$, initial class balance degree $d_p \leftarrow -n/(P-1)$, where $p = 1, \ldots, P$.
Output: $\mathbf{Y}$.

```
1: Sort $\mathbf{c}$ in the *ascend* order, denoted as $\mathbf{c}'$. Align $\mathbf{z}$ in the
   same sequence with the sorted $\mathbf{c}$, denoted as $\mathbf{z}'$.
2: for $i = 1, \ldots, nP$ do
3:   if $m_i == 0$ then
4:     $a \leftarrow 0$
5:     Find the corresponding element of $z'_i$ in the original $\mathbf{Y}$.
       Suppose $z'_i$ locates in the $j$th row (sample) and the
       $p$th column (class) of the original $\mathbf{Y}$.
6:     if $d_p < -l/(P-1)$ then
7:       $a \leftarrow 1$
8:     else if $-l/(P-1) \le d_p \le l$ then
9:       if $O + c'_i(1 + 1/(P-1)) < O$ then
10:        $a \leftarrow 1$
11:      end if
12:    end if
13:    if $a == 1$ then
14:      $\bar{y}_{j,p} \leftarrow 1$
15:      $O \leftarrow O + c'_i(1 + 1/(P-1))$
16:      $d_p \leftarrow d_p + (1 + 1/(P-1))$
17:      Find $P$ corresponding elements of $\bar{\mathbf{y}}_j$ in $\mathbf{z}'$.
         Suppose these elements locates in the $j_1, \ldots, j_P$
         row of $\mathbf{z}'$; then, set $m_{j_t} = 1$ where $t = 1, \ldots, P$,
         so as to tell the procedure that these elements should
         not be handled again.
18:    end if
19:  end if
20: end for
```

VII. Discussion: Why Do We Use the Regression Approach?

The most important reason of utilizing the regression approach is to prevent solving an integer matrix programming problem. Here, we only analyze the binary-class case. The same phenomenon can be observed in the multiclass case as well.

As we know, the time complexity of an algorithm is determined by its most computationally expensive part. If we take the standard SVM as the core of the MMC (6), we would get a new MMC objective defined as

$$\min_{\mathbf{y}\in\mathcal{B}_0} \left\{ \min_{\mathbf{w},\boldsymbol{\xi}\ge\mathbf{0}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^n \xi_i \quad \text{s.t. } y_i \mathbf{w}^T \phi(\mathbf{x}_i) \ge 1 - \xi_i \qquad \forall i = 1, \ldots, n \right\}. \tag{52}$$

If we rewrite the problem in the brackets of (52) in its dual, problem (52) is equivalent to the following problem:

$$\min_{\mathbf{M}\in\mathcal{Y}_0} \left\{ \max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T \mathbf{1} - \frac{1}{2}\boldsymbol{\alpha}^T (\mathbf{K} \circ \mathbf{M}) \boldsymbol{\alpha} \quad \text{s.t. } 0 \le \alpha_i \le \frac{C}{n} \qquad \forall i = 1, \ldots, n \right\} \tag{53}$$

where operator $\circ$ denotes the element-wise product and set $\mathcal{Y}_0$ is defined as $\mathcal{Y}_0 = \{\mathbf{M} | \mathbf{M} = \mathbf{y}\mathbf{y}^T, \forall \mathbf{y} \in \mathcal{B}_0\}$. It is obvious that problem (53) is a binary integer matrix programming problem. Generally, solving the problem is *NP-complete*.

Although Xu *et al.* [16] relaxed the integer matrix to a matrix with continuous values, the relaxed problem is reformulated as an SDP problem with $\mathcal{O}(n^2)$ parameters that is solved in time $\mathcal{O}(n^3)$ or higher. However, if we use the SVR as the core, the parameter number will be lowered to $\mathcal{O}(n)$.

Although Li *et al.* [41] solves the problem by utilizing the fact that $\mathbf{M} = \mathbf{y}\mathbf{y}^T$, they encounter a problem that the expressions of the samples in the kernel-induced feature space should be explicitly gotten by first calculating and storing the kernel matrix $\mathbf{K}$ with $\mathcal{O}(n^2)$ time and storage complexities, respectively, and then decomposing the kernel matrix [35], [37] with an additional time of at least $\mathcal{O}(n^2)$ [38], [39]. This is also impractical for large-scale problems. However, if the SVR is utilized, the kernel decomposition can be avoided.

Summarizing the aforementioned, we should try to avoid operating on $\mathbf{M}$ in the study of MMC. The SVR provides us this feasibility. In fact, although the SVR was originally proposed to estimate linear regression problems, it can be used to construct both binary class classifiers and multiclass classifiers [80], [85].

VIII. Theoretical Analysis

In this section, we will analyze the computational and storage complexities of Algorithms 1, 3, and 5.

A. Computational Complexity

Algorithms 1, 3, and 5 contain two loops, i.e., the outer cutting-plane loop and the inner ELM loop. Suppose each algorithm will need an average of T_c outer cutting-plane iterations to converge to a global optimum. Suppose each cutting-plane subproblem will need an average of T_e inner ELM iterations. In each ELM iteration, the algorithms need to call the *SKSVR* function (Algorithm 2) or the *MKSVR* function (Algorithm 4). Suppose the *SKSVR* function or the *MKSVR* function needs an average of T_s iterations to converge. Suppose the $\mathrm{estimate\_basis}$ function in *SKSVR* and *MKSVR* needs an average of r iterations to converge. We can get the following theorem.

Theorem 9: When the data set is large scale, Algorithms 1, 3, and 5 have linearthmic time complexities of $\mathcal{O}(T_cT_eT_srsn + T_cn\log n)$, $\mathcal{O}(QT_cT_eT_srsn + T_cn\log n)$, and $\mathcal{O}(PT_cT_eT_srsn + T_cPn\log Pn)$, respectively, where s is the sparsity of the data set.

Proof: We first analyze the complexity of the SVR-MMC (Algorithm 1) in the first paragraph. Next, we analyze the time complexities of the SVR-MKC (Algorithm 3) and the SVR-M3C (Algorithm 5) in the second paragraph. The following proof is under the assumption that the data set is large scale.

For ***Layer 3*** or the *SKSVR* function, the number of the basis vectors is irrelevant to the data set size n and usually far smaller than n [84], so as to the number of the unknown parameters. It means that the time complexity of the *SKSVR* function is mainly determined by the $\mathrm{estimate\_basis}$ function. Because each call of the *SKSVR* function will call the $\mathrm{estimate\_basis}$ function T_s times, we can conclude from Section IV-D2 that the time complexity of the *SKSVR* function is about $\mathcal{O}(T_srsn)$. For ***Layer 2***, a single cutting-plane iteration needs T_e ELM iterations and calculate the most violated $\mathbf{y}$ once. Each ELM iteration needs to call *SKSVR* once and update $\boldsymbol{\mu}$ by calling the *projection* function. Because the *projection* function is irrelevant to n, its time complexity can be omitted. From Theorem 1, calculating the most violated $\mathbf{y}$ needs $\mathcal{O}(n\log n)$ time. Therefore, a cutting-plane iteration needs about $\mathcal{O}(T_eT_srsn + n\log n)$ time. For ***Layer 1***, the SVR-MMC needs T_c outer cutting-plane iterations. As a conclusion, the time complexity of the SVR-MMC is about $\mathcal{O}(T_cT_eT_srsn + T_cn\log n)$.

Because the most significant difference in time complexity between the SVR-MMC and the SVR-MKC is that the time complexity of the *MKSVR* function is Q times as high as that of the *SKSVR*. Hence, the time complexity of the SVR-MKC is about $\mathcal{O}(QT_cT_eT_srsn + T_cn\log n)$. Analogously, the most significant difference between the SVR-MMC and the SVR-M3C is that the SVR-M3C calls the *SKSVR* functions P times in a single ELM iteration, whereas the SVR-MKC only calls the *SKSVR* once. Also, calculating the most violated $\mathbf{Y}$ (Algorithm 6) consumes $\mathcal{O}(Pn\log Pn)$. Therefore, the time complexity of the SVR-M3C is $\mathcal{O}(PT_cT_eT_srsn + T_cPn\log Pn)$. ■

B. Storage Complexity

We first analyze the storage complexity of the SVR-MMC (Algorithm 1) in the first paragraph. Next, we analyze the storage complexities of the SVR-MKC (Algorithm 3) and the SVR-M3C (Algorithm 5) in the second paragraph.

First of all, we need to store the whole data set, which requires an $\mathcal{O}(snd)$ space. For each outer cutting-plane iteration, we need to store a new violated constraint $\mathbf{y}$, which requires an $\mathcal{O}(n)$ space. For each inner ELM iteration, we need to store a new vector $\mathbf{G}\boldsymbol{\alpha}$, which also requires an $\mathcal{O}(n)$ space. For each call of the *SKSVR* function, we need to store the most violated constraint set, which occupies another $\mathcal{O}(T_sn)$ space. Because there are few basis vectors, the space for the basis vectors and the (sparse) base kernel matrix can be omitted. Therefore, the storage complexity of the SVR-MMC is about $\mathcal{O}(snd + (T_c + T_cT_e + T_s)n)$, which is linear with n.

For the SVR-MKC, because the space for the basis vectors can be omitted, the SVR-MKC (Algorithm 3) has **the same storage complexity** as the SVR-MKC. For the SVR-M3C, we need to store a new violated constraint $\mathbf{Y}$ in each outer cutting-plane iteration, which requires an $\mathcal{O}(Pn)$ space. We also need to store P new vectors $\{\mathbf{G}_p\boldsymbol{\alpha}_p\}_{p=1}^P$ in each inner ELM iteration, which requires another $\mathcal{O}(Pn)$ space. Hence, the storage complexity of the SVR-M3C (Algorithm 5) is about $\mathcal{O}(snd + (PT_c + PT_cT_e + T_s)n)$.

Summarizing the aforementioned, the storage complexities of the three proposed algorithms are all linear with the data set size.

IX. Experimental Analysis

In this section, we will evaluate our SVR-MMC, SVR-MKC, and SVR-M3C algorithms on effectiveness and efficiency

TABLE I
DESCRIPTIONS OF THE DATA SETS. "n" IS THE DATA SET SIZE, "d" IS THE DIMENSION, AND "s" IS THE SPARSITY

ID	Data	Size (n)	Feature (d)	Class (P)	Sparsity (s)
1	Echocardiog	132	8	2	96.95
2	Spectf	267	44	2	100.00
3	Heart-stalog	270	13	2	75.10
4	Haberman	306	3	2	85.19
5	LiveDisorders	345	6	2	99.57
6	Ionosphere	351	34	2	88.09
7	House-votes	435	16	2	100.00
8	Clean1	476	166	2	99.84
9	Australian	690	14	2	93.90
10	Breast	699	9	2	99.75
11	Diabetes	768	8	2	95.38
12	German	1000	24	2	74.95
13	Satellite1vs2	1551	36	2	100.00
14	LetterAvsB	1555	16	2	98.89
15	Abalone1vs2	2835	8	2	100.00
16	Krvskp	3196	36	2	100.00
17	Waveform1vs2	3308	40	2	99.70
18	Spambase	4601	57	2	71.51
19	MNIST1vs7	15170	784	2	13.75
20	MNIST2vs7	14283	784	2	19.10
21	MNIST3vs8	13966	784	2	21.48
22	MNIST8vs9	13783	784	2	20.20
23	Covtype1vs2	595141	54	2	22.01
24	Dermathology	366	34	6	40.37
25	Iris	150	4	3	100.00
26	Balance	625	4	3	100.00
27	Shuttle	14500	9	7	78.27
28	Libras1-7	168	90	7	99.98
29	Synthetic	600	60	6	100.00
30	OptDigits03689	2800	64	5	52.96
31	OptDigits12457	2820	64	5	49.37
32	PenDigits03689	5364	16	5	87.22
33	PenDigits12457	5628	16	5	87.13
34	20-newsgroup-rec	3979	26214	4	0.32
35	ShortMessage	42095	87	3	2.58

empirically. Specifically, we will first compare the three algorithms with several clustering methods on various real-world data sets. Then, we will illustrate the scaling behaviors of the SVR-MMC and SVR-MKC algorithms. At last, we will apply the SVR-MKC algorithm to the VAD [86], [87]. All experiments are conducted with MATLAB 7.12 on a 2.4-GHz Intel Core 2 Duo personal computer running Windows XP with 6-GB main memory.

A. Data Sets

The experiments are performed on 35 data sets. The detailed information of the data sets are listed in Table I. Except the data sets from the **MNIST** handwritten digit database,[4] data set from **20 − newsgroup** text database[5] and our **ShortMessage** text data set from China Mobile company, all other data sets are broadly selected from the UCI machine learning repository.[6]

In respect of the binary-class problems, because there are multiple classes in the **Satellite**, **Letter**, **Abalone**, **Waveform**, and **Covtype** data sets, we use their first two classes only (1 versus 2, A versus B, 1 versus 2, 1 versus 2, and 1 versus 2, respectively). For the **MNIST** data set, we follow [25] and [41], and focus on the digital pairs that are difficult to differentiate, i.e., 1 versus 7, 2 versus 7, 3 versus 8, and 8 versus 9.

In respect of the multiclass problems, we conduct experiments on the first seven classes of **Libras**. For the **OptDigits** and **PenDigits** data sets, we partition the digits that are easily confused with each other to the same group. For

[4]http://yann.lecun.com/exdb/mnist.
[5]http://people.csail.mit.edu/jrennie/20Newsgroups.
[6]All UCI data sets are downloaded from http://archive.ics.uci.edu/ml.

20 − newsgroup, we follow [27] and choose topic **rec**, which contains *autos*, *motorcycles*, *baseball*, and *hockey*.

B. Experimental Settings

For the SVR-MMC algorithm, the cutting-plane solution precision η is set to 0.01, the ELM solution precision τ is set to 0.0005, and the CPSP solution precision of the *SKSVR* function ϵ is set to 0.001. Parameter l is searched through $\{0, 0.03n, 0.1n, 0.2n, 0.3n, 0.5n, 0.7n\}$. In this paper, the linear, polynomial, and RBF kernels are used for performance analysis. The SVR-MMC algorithms with the linear, polynomial, and RBF kernels are denoted as SVR-MMC$_l$, SVR−MMC$_p$, and SVR-MMC$_r$, respectively. For SVR-MMC$_l$, parameter C is searched from the exponential grid $2^{[12:1:27]}$. For SVR-MMC$_p$, parameter C is searched from $2^{[-2:1:9]}$; the kernel parameter p of the polynomial kernel is searched through {2, 3, 4}. For SVR-MMC$_r$, parameter C is also searched from $2^{[-2:1:9]}$; the RBF kernel width σ is searched through $\{0.25\gamma, 0.5\gamma, \gamma, 2\gamma, 4\gamma\}$, where γ is the average Euclidean distance between the samples.

For the SVR-MKC algorithm, parameter C is searched from $2^{[-2:1:9]}$. The linear, polynomial, and RBF kernels are used as three base kernels. The parameters of the base kernels are searched in the same way as the SVR-MMCs. All other parameters are set to the same values as the SVR-MMC.

For the SVR-M3C algorithm, the cutting-plane solution precision η is set to 0.1. The linear and RBF kernels are used for performance analysis. The kernel parameters are searched in the same way as the SVR-MMCs. All other parameters are set to the same values as the SVR-MMC.

To examine the effectiveness of the proposed three algorithms, we compare them with our types of data-clustering methods.

1) Classic clustering methods:
   - K-means (KM) [4]. We run it 40 times and report the average results.[7]
   - Normalized cut (NC) [9].
2) Recently reported MMC methods:[8]
   - IterSVR [24], [25].[9] We run it 20 times and report the average experimental results.
   - CPMMC [26], [27].[10]
   - LG-MMC [41].[11] The maximum cutting-plane iteration number is set to 20 when $n < 10\,000$ and to 30 when $n \geq 10\,000$ according to the experimental conclusions of Li *et al.* When the data sets are large scale, the linear kernel is used.
3) CPMKC [40]. We implemented the algorithm by modifying the CPMMC [26], [27] code. The SOCP problem of the CPMKC is solved by the *fmincon*} function in MATLAB. The kernel selection scheme is the same as [40]. Parameters C and l are searched in the same way as the CPMMC algorithm. According to [40], the KPCA [35] is used as the interface to the nonlinear base kernels.[12] Since the authors of [40] did not mention the parameter selection schemes of the base kernels, the parameters of the base kernels are searched in the same way as our SVR-MKC algorithm.
4) CPM3C [49]. We implemented the algorithm by modifying the CPMMC code. All parameters are searched in the same way as the CPMMC algorithm.

For each data set, we run the algorithms once and report the best achievable performances. The samples are normalized into the range of [0, 1] in dimension [88]. All computation times are recorded except that consumed on normalizing the data set. Note that, for the KM and the IterSVR, the averages of the best performances are reported, and all experiments are exactly run with **the authors' experimental settings**.

C. Evaluation Results

We do the evaluations on binary-class problems and multiclass problems separately.

1) Results on Binary-Class Problems: The clustering accuracies of the SVR-MMC, the SVR-MKC, and other referenced methods are listed in Table II. From the table, we can clearly see that the SVR-MMCs and the SVR-MKC are superior to the referenced clustering algorithms. From the table, we can also know that the SVR-MKC algorithm can automatically select the suitable kernels for the robustness of the SVR-MMC.

The CPU time comparison of our algorithms and the referenced methods are reported in Table III. From the table, we can observe the following: 1) the SVR-MMC, SVR-MKC, KM, and CPMMC algorithms are relatively fast; 2) the CPU time of NC, IterMMC, LG-MMC, and CPMKC on KPCA dramatically grows with n; 3) although the CPU time of CPMKC on clustering does not show an explicit relation with n, it is obviously slower than the proposed SVR-MKC algorithm. Therefore, the SVR-MMC and SVR-MKC algorithms are efficient. From the table, we can also know that, because the SVR-MMCs and the SVR-MKC are solved globally via GELM, their cutting-plane iteration numbers are usually small. Finally, we can get guaranteed clustering accuracies in guaranteed clustering time. Note that, because the basis vector estimation algorithm for the polynomial kernel is relatively slow, it should be further improved in the future.

2) Results on Multiclass Problems: The clustering accuracies of the SVR-M3C and the referenced multiclass clustering methods are listed in Table IV. From the table, we can see that

[7]The implementation code is in the VOICEBOX developed by Cambridge University for speech processing. It can be downloaded from http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/doc/voicebox/kmeans.html.

[8]Because the MMC proposed in [16], the GMMC proposed in [21], and the M3C proposed in [22] are based on SDP and can be only run with at most hundreds of samples, we will not compare with them.

[9]The implementation code can be downloaded from http://www3.ntu.edu.sg/home/IvorTsang/itMMC_code.zip.

[10]The implementation code can be downloaded from http://sites.google.com/site/binzhao02.

[11]The implementation code can be downloaded from http://cs.nju.edu.cn/zhouzh/zhouzh.files/publication/annex/LGMMC_v2.rar.

[12]The implementation code is in the SVM-KM toolbox http://asi.insa-rouen.fr/enseignants/~arakotom/toolbox/index.html.

TABLE II
ACCURACY (IN PERCENT) COMPARISON OF DIFFERENT CLUSTERING METHODS ON **BINARY-CLASS PROBLEMS**. ROW **RANK** IS THE AVERAGE RANKS OF THE METHODS OVER THE FIRST 18 DATA SETS

ID	Data	KM	NC	IterSVR	CPMMC	LG-MMC	CPMKC	SVR-MMC$_l$	SVR-MMC$_p$	SVR-MMC$_r$	SVR-MKC
1	Echocardiogram	87.18	93.13	90.23	93.89	86.26	89.31	87.02	93.13	**94.66**	**94.66**
2	Spectf	62.68	75.28	61.35	79.40	77.15	63.30	80.16	79.40	**83.90**	80.52
3	Heart-stalog	74.39	79.63	75.07	80.00	60.37	70.37	78.15	78.15	**82.22**	77.41
4	Haberman	51.23	51.96	66.67	73.53	73.86	73.53	73.53	73.86	74.18	**76.14**
5	LiveDisorders	54.62	57.10	56.80	57.97	**67.54**	60.29	58.55	63.19	62.03	65.22
6	Ionosphere	71.03	70.37	71.60	70.37	73.79	77.21	**81.77**	75.78	79.77	78.06
7	House-votes	84.09	88.51	84.17	85.98	85.29	**94.48**	84.83	86.21	84.14	89.20
8	Clean	50.63	51.05	52.16	71.64	68.28	93.07	95.59	93.07	**99.79**	98.53
9	Australian	85.38	89.71	83.93	**96.38**	66.52	66.67	69.71	84.78	95.22	95.51
10	Breast	94.93	95.99	95.56	96.57	77.68	96.85	85.41	96.14	96.71	**97.00**
11	Diabetes	95.33	91.93	95.83	90.89	93.23	**96.87**	92.19	93.37	92.97	95.70
12	German	60.59	59.50	60.60	70.00	70.70	70.60	70.10	70.80	70.60	**71.70**
13	Satellite1vs2	95.62	97.29	95.87	86.40	69.12	**98.97**	98.19	97.03	97.16	98.65
14	LetterAvsB	93.70	90.74	92.63	93.50	80.90	94.47	82.89	89.07	92.09	**95.18**
15	Abalone1vs2	52.72	52.95	52.82	53.90	55.38	54.07	53.90	55.98	55.45	**56.83**
16	Krvskp	54.56	50.88	54.63	55.04	60.73	55.44	59.42	65.24	**70.43**	70.02
17	Waveform1vs2	94.48	94.53	94.45	94.47	63.51	**95.13**	94.44	95.07	94.86	95.07
18	Spambase	**100.00**	60.62	**100.00**	**100.00**	73.79	78.18	99.04	**100.00**	**100.00**	**100.00**
Rank		7.5556	6.9444	6.9444	5.3889	6.9444	4.5556	6.0000	3.9444	3.2222	1.9444
19	MNIST1vs7	95.77	–	–	96.74	96.88	–	97.72	95.91	**98.98**	98.53
20	MNIST2vs7	95.56	–	–	96.04	93.32	–	94.43	95.66	**97.75**	95.98
21	MNIST3vs8	79.94	–	–	86.10	88.66	–	88.41	90.22	**95.36**	94.75
22	MNIST8vs9	80.10	–	–	54.13	90.03	–	81.78	91.71	**96.98**	95.01
23	Covtype1vs2	51.15	–	–	57.22	–	–	57.22	66.72	**69.00**	–

TABLE III
CPU TIME (IN SECONDS) AND ITERATION NUMBERS (IN THE BRACKETS) OF DIFFERENT CLUSTERING METHODS ON **BINARY-CLASS PROBLEMS**

ID	Data	KM	NC	IterSVR	CPMMC	LG-MMC	CPMKC		SVR-MMC$_l$	SVR-MMC$_p$	SVR-MMC$_r$	SVR-MKC
							KPCA	Clustering				
1	Echocardiogram	0.00	0.03	0.21 (4.2)	0.11 (3)	2.16 (20)	0.07	499.60 (30)	0.14 (2)	1.50 (3)	1.72 (9)	2.53 (2)
2	Spectf	0.02	0.15	0.58 (3.0)	0.03 (2)	7.23 (20)	0.41	978.67 (30)	0.32 (2)	2.64 (4)	5.06 (4)	5.56 (5)
3	Heart-stalog	0.01	0.11	0.68 (2.8)	0.02 (2)	7.88 (20)	0.35	921.90 (30)	1.03 (3)	1.30 (2)	0.92 (3)	5.00 (5)
4	Haberman	0.00	0.13	1.31 (8.7)	0.02 (2)	9.67 (20)	0.55	20.40 (2)	0.49 (2)	0.72 (2)	1.14 (3)	15.80 (16)
5	LiveDisorders	0.01	0.18	1.68 (9.4)	0.03 (2)	12.38 (20)	0.66	11.13 (2)	0.12 (2)	1.31 (2)	1.52 (3)	3.73 (3)
6	Ionosphere	0.02	0.19	2.28 (4.0)	0.05 (2)	13.67 (20)	0.70	1466.22 (30)	2.26 (4)	4.43 (3)	3.14 (8)	1.93 (3)
7	House-votes	0.01	0.25	2.22 (2.7)	0.10 (3)	26.08 (20)	1.08	366.74 (5)	0.09 (2)	5.08 (3)	2.72 (3)	18.74 (8)
8	Clean	0.20	0.51	4.48 (4.6)	0.04 (2)	30.97 (20)	1.97	749.98 (12)	1.04 (2)	62.24 (3)	3.45 (3)	102.88 (8)
9	Australian	0.03	0.72	12.18 (7.9)	0.11 (3)	61.55 (20)	3.30	157.18 (2)	1.14 (2)	6.55 (4)	3.75 (3)	17.55 (11)
10	Breast	0.01	0.70	11.16 (4.5)	0.03 (2)	62.95 (20)	3.70	300.41 (11)	0.64 (2)	8.46 (6)	0.80 (2)	28.53 (25)
11	Diabetes	0.02	0.86	13.75 (3.0)	0.06 (2)	76.73 (20)	4.64	545.45 (30)	0.45 (3)	3.24 (2)	0.35 (2)	37.08 (30)
12	German	0.03	1.65	1.93 (2.6)	0.04 (2)	124.08 (20)	11.50	560.94 (30)	0.48 (2)	1.69 (2)	1.17 (2)	8.68 (4)
13	Satellite1vs2	0.09	3.93	13.98 (4.0)	0.10 (2)	294.92 (20)	39.96	1424.92 (30)	1.08 (3)	8.25 (3)	1.17 (2)	18.31 (4)
14	LetterAvsB	0.04	3.97	39.48 (4.0)	0.13 (2)	289.52 (20)	55.31	256.72 (5)	2.07 (3)	3.04 (2)	2.38 (4)	7.15 (2)
15	Abalone1vs2	0.10	12.85	27.59 (4.2)	0.10 (2)	889.91 (20)	233.55	182.12 (6)	0.36 (2)	15.73 (2)	1.07 (2)	20.39 (2)
16	Krvskp	0.41	17.82	724.72 (6.9)	0.09 (2)	1224.59 (20)	339.15	70.10 (3)	1.60 (2)	44.47 (3)	22.74 (7)	291.20 (14)
17	Waveform1vs2	0.84	20.47	291.49 (2.4)	0.17 (2)	1246.75 (20)	380.76	183.76 (3)	0.95 (4)	40.94 (2)	5.27 (5)	34.07 (4)
18	Spambase	0.26	20.90	4.78 (2.0)	0.24 (3)	2496.14 (20)	1042.96	2716.42 (30)	1.87 (3)	31.84 (3)	2.86 (3)	29.37 (3)
19	MNIST1vs7	43.89	–	–	15.74 (4)	16576.73 (31)	–	–	13.00 (3)	650.74 (2)	238.12 (3)	3824.46 (4)
20	MNIST2vs7	48.03	–	–	2.74 (3)	23621.14 (30)	–	–	39.81 (27)	3768.91 (8)	417.87 (3)	3339.20 (6)
21	MNIST3vs8	80.75	–	–	7.48 (4)	24790.30 (31)	–	–	15.00 (2)	1903.10 (4)	306.55 (3)	2625.30 (5)
22	MNIST8vs9	106.23	–	–	43.43 (4)	40429.27 (30)	–	–	14.08 (4)	2545.30 (5)	642.05 (3)	5062.45 (7)
23	Covtype1vs2	30.19	–	–	46.33 (2)	–	–	–	2423.70 (2)	4218.10 (3)	4772.76 (4)	–

the SVR-M3C can achieve higher clustering accuracies than the referenced clustering algorithms.

The CPU time comparison is also reported in Table V. From the table, we can also see that the proposed SVR-M3C has a comparable efficiency with KM and CPM3C, which have linear time complexities.

D. *Study of the Scaling Behavior*

Here, we focus on the scaling behavior of the proposed algorithms on binary-class problems. The scaling behavior of the proposed algorithms on the multiclass problems is similar with that on the binary-class problems.

TABLE IV
ACCURACY (IN PERCENT) COMPARISON OF DIFFERENT CLUSTERING METHODS ON **MULTICLASS PROBLEMS**

ID	Data	KM	NC	CPM3C	SVR-M3C$_l$	SVR-M3C$_r$
24	Dermathology	70.38	**75.14**	63.93	50.82	71.86
25	Iris	81.62	66.00	72.67	84.67	**88.00**
26	Balance	52.50	50.72	75.52	**88.16**	87.68
27	Shuttle	41.75	–	82.01	86.97	**87.15**
28	Libras	51.76	63.10	26.19	**75.60**	74.40
29	Synthetic	**63.06**	42.83	33.33	51.33	61.67
30	OptDigits03689	79.99	76.93	79.14	59.71	**93.54**
31	OptDigits12457	86.88	81.49	79.75	85.25	**88.87**
32	PenDigits03689	**74.97**	–	69.30	63.78	65.44
33	PenDigits12457	**67.19**	–	58.23	63.79	40.44
34	20-newsgroup-rec	26.42	–	–	**89.19**	82.23
35	ShortMessage	41.52	–	66.22	80.03	**81.38**

TABLE V
CPU TIME (IN SECONDS) AND ITERATION NUMBERS (IN THE BRACKETS) OF DIFFERENT CLUSTERING METHODS ON **MULTICLASS PROBLEMS**

ID	Data	KM	NC	CPM3C	SVR-M3C$_l$	SVR-M3C$_r$
24	Dermathology	0.14	0.10	0.65 (2)	295.06 (8)	21.16 (7)
25	Iris	0.01	0.02	57.79 (30)	0.88 (3)	16.99 (20)
26	Balance	0.03	0.24	0.16 (2)	45.87 (15)	0.89 (3)
27	Shuttle	2.59	–	828.89 (10)	15.31 (3)	205.23 (2)
28	Libras	0.19	0.04	56.51 (7)	5.35 (2)	2.85 (2)
29	Synthetic	0.33	0.36	21.07 (6)	6.81 (5)	21.17 (5)
30	OptDigits03689	1.09	7.00	3.47 (2)	17.35 (3)	2.45 (2)
31	OptDigits12457	1.92	5.93	5.71 (2)	10.48 (4)	2.49 (2)
32	PenDigits03689	1.02	–	4.60 (2)	3.48 (2)	78.00 (5)
33	PenDigits12457	1.02	–	3.90 (2)	1.67 (2)	131.46 (7)
34	20-newsgroup-rec	2574.53	–	–	104.95 (2)	1158.08 (2)
35	ShortMessage	6.02	–	112.48 (2)	13.69 (3)	1538.85 (3)

The UCI **Adult** data set is used for this study. We use the UCI **Adult** data set[13] in a way that is similar to [24]. More precisely, serial subsets of the **Adult** data set are predefined, ranging in size [1605, 2265, 3185, 4781, 6414, 11 220, 16 100, 22 696, 32 561]. Because the data set is severely imbalance, Zhang *et al.* [24] used the *balanced clustering error* as the metric. For consistent comparison with their work [24], we report the performance of the proposed algorithm in this metric as well. The balanced clustering error rate Err_B is defined as $\mathrm{Err}_B = (\mathrm{Err}^+ + \mathrm{Err}^-)/2$, where Err^+ and Err^- are the error rates of the positive and negative samples in the full set.

Experimental results are shown in Fig. 2. The empirical time complexities of the various methods are summarized in Table VI. From the figure and the table, we can conclude that the KM, CPMMC, SVR-MMC, and SVR-MKC algorithms have linear time complexities, whereas the IterMMC, LG-MMC, and CPMKC algorithms have time complexities of at least $\mathcal{O}(n^2)$. The reason why the empirical time complexities of the SVR-MMC and SVR-MKC algorithms are linear but not linearithmic is that, we think, the unit of the real CPU time spent on the linearithmic time complexity part $\mathcal{O}(n \log n)$ is shorter than that spent on the linear time complexity part $\mathcal{O}(tsn)$.

[13]http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/.

E. Application to VAD

In this subsection, we apply the SVR-MMC and SVR-MKC algorithms to the VAD [86], [87]. VAD tries to discriminate speech against the background noise. It is one of the key issues in practical speech systems. For instance, it is used as the front-end of large-vocabulary continuous-speech recognition systems. By eliminating the unvoiced signals, the recognition rate can be improved. It is used as the front-end of modern speech communication systems. By filtering out the unvoiced signals, the band efficiency of the communication can be increased.

Currently, machine-learning-based VAD methods are hot. Typically, a machine-learning-based approach can be partitioned into two parts. The first part is to extract acoustic features from noisy speeches. The second part is to use a binary-class classifier to discriminate the acoustic features. There are various acoustic features that are suitable for machine-learning-based approaches [89]–[97].

In this paper, we focus on classifiers but not acoustic features. In respect of classifiers, the machine-learning-based VAD meets the following three challenges: 1) How to achieve robust performance in low signal-to-noise ratio (SNR) and nonstationary noisy environments? 2) How to alleviate the labeling requirement, since the labeling might be inaccurate in low

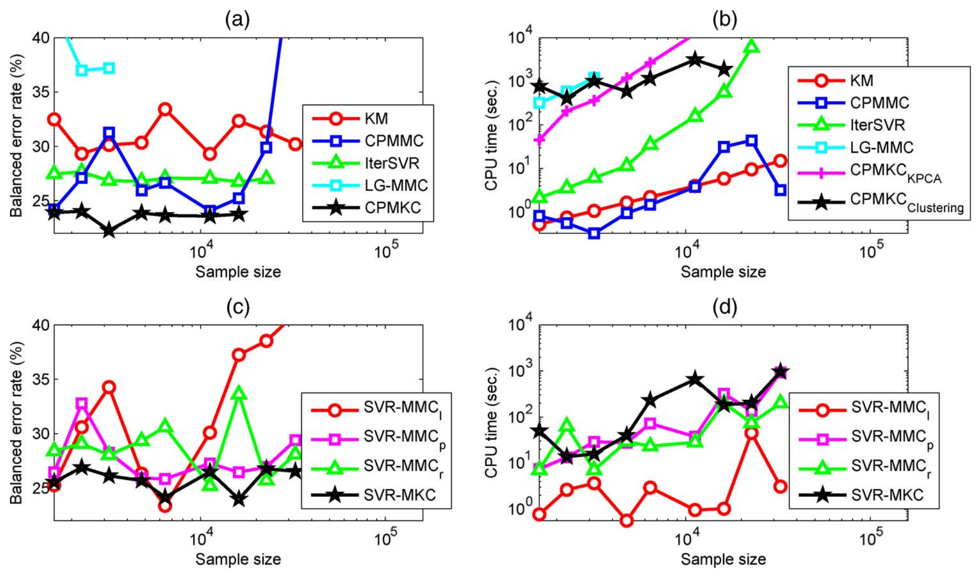

Fig. 2. Balanced error rate (in percent) and CPU time (in seconds) comparisons of the clustering algorithms on the **Adult** subsets. (a) and (b) are on the referenced methods. (c) and (d) are on the proposed methods.

TABLE VI
EMPIRICAL TIME COMPLEXITIES OF VARIOUS ALGORITHMS ON THE **Adult** DATA SET

KM	IterSVR	CPMMC	LG-MMC	CPMKC		SVR-MMC$_l$	SVR-MMC$_p$	SVR-MMC$_r$	SVR-MKC
				KPCA	Clustering				
$\mathcal{O}(n^{1.16})$	$\mathcal{O}(n^{2.69})$	$\mathcal{O}(n^{1.35})$	$\mathcal{O}(n^{1.94})$	$\mathcal{O}(n^{2.14})$	$\mathcal{O}(n^{0.54})$	$\mathcal{O}(n^{0.53})$	$\mathcal{O}(n^{1.42})$	$\mathcal{O}(n^{0.94})$	$\mathcal{O}(n^{1.27})$

SNR environments and the costs on manual labeling are very expensive? 3) How to meet the strong real-time detection demand, since VAD usually works online?

Recently, the multiple-feature-based data fusion methods have been tried to beat the first challenge [89], [93], [94], [96]–[98]. The unsupervised learning methods were tried to beat the second challenge [96], [99], [100]. An efficient MKL method was proposed to beat the third challenge [98].

However, these works cannot meet the three requirements simultaneously. Here, we propose to apply the SVR-MKC to VAD for all of the aforementioned three challenges. Theoretically, aside from the convexity, the SVR-MKC can hold multiple features, needs no labeling for model training, and has an $\mathcal{O}(n \log n)$ training complexity and an $\mathcal{O}(1)$ prediction complexity for the real-time demand. To our knowledge, this is the first work that beat all of these challenges simultaneously.

To better show the advantages of the SVR-MKC-based VAD, we follow the experimental settings of [98] and conduct the following experiments.

Seven noisy test corpora of the AURORA2 [101] are used. The AURORA2 is an open corpus that has been widely used in speech recognition and VAD. The SNR level is about 5 dB. Each test corpus contains 1001 utterances, which are randomly split into three groups for unsupervised training, developing, and evaluation, respectively. Each training set and development set consists of 300 utterances, respectively. Each evaluation set consists of 401 utterances. We concatenate all short utterances in each data set into a long one so as to simulate the real-world application environment of VAD. Eventually, the length of each long utterance is in a range of (450, 750) s with about 65% speeches. The observation (sample) is 25 ms long with an overlap of 15 ms.

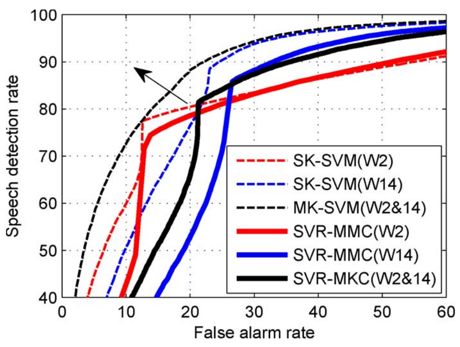

Fig. 3. ROC curve comparison of the SVR-MKC- and MK-SVM-based VADs in car noise (SNR $= 5$ dB). "W#" are short for the MO-MP features with different window lengths. "SK-SVM" is short for the single-kernel SVM.

The supervised multiple kernel (MK)-SVM [98] is used for comparison. The multiple-observation maximum probability (MO-MP) features [96] are used for performance analysis.

The receiver-operating-characteristic (ROC) curve [102] is considered as a general performance measurement of the VAD. VAD usually works on a point of the ROC curve, which is called the *operating point*. As shown in Fig. 3, the closer to the upper left corner the ROC curve is, the better the VAD performs. Because the MO-MP features with different window lengths yield different ROC curves, they can be seen as different feature expressions. We use two MO-MP features with window lengths of 2 and 14 as the inputs of the SVR-MKC and MK-SVM algorithms.

TABLE VII
ACCURACY COMPARISON (IN PERCENT) OF THE SVR-MKC- AND MK-SVM-BASED VADs IN SEVEN NOISE SCENARIOS. "**W#**" IS SHORT FOR THE MO-MPs WITH DIFFERENT WINDOW LENGTHS. THE VALUES IN BRACKETS ARE STANDARD DEVIATIONS

Noise	**SK-SVM(W2)**	**SK-SVM(W14)**	**MK-SVM(W2&14)**
Babble	56.86 (5.67)	73.70 (1.70)	**75.04** (0.85)
Car	81.67 (0.48)	83.57 (0.25)	**84.60** (0.16)
Restaurant	71.13 (1.07)	73.39 (1.57)	**73.94** (0.88)
Street	55.17 (1.08)	**62.06** (5.70)	61.40 (3.80)
Airport	73.79 (0.45)	73.70 (0.58)	**74.70** (0.49)
Train	72.89 (1.80)	73.90 (1.39)	**75.77** (0.61)
Subway	71.14 (1.17)	**76.28** (1.37)	75.77 (1.57)
Noise	**SVR-MMC(W2)**	**SVR-MMC(W14)**	**SVR-MKC(W2&14)**
Babble	60.30 (5.70)	**73.86** (0.79)	73.81 (0.38)
Car	77.40 (0.92)	80.39 (0.94)	**80.71** (0.74)
Restaurant	68.26 (1.63)	**70.38** (0.18)	69.76 (0.65)
Street	55.30 (1.59)	60.15 (2.47)	**63.05** (3.61)
Airport	68.03 (2.28)	67.86 (3.31)	**68.46** (1.04)
Train	71.79 (0.89)	70.67 (2.77)	**72.91** (2.48)
Subway	**68.24** (1.79)	67.10 (3.04)	67.83 (2.45)

TABLE VIII
CPU TIME (IN SECONDS) OF THE SVR-MKC-BASED VADs ON PREDICTION. EACH TEST SET IS ABOUT [450, 750] s LONG

Noise	**SVR-MMC(W2)**	**SVR-MMC(W14)**	**SVR-MKC(W2&14)**
Babble	0.55 (0.07)	0.62 (0.10)	1.04 (0.13)
Car	0.57 (0.04)	0.58 (0.05)	1.00 (0.08)
Restaurant	0.68 (0.11)	0.54 (0.07)	1.24 (0.22)
Street	0.64 (0.05)	0.67 (0.14)	1.27 (0.31)
Airport	0.58 (0.05)	0.54 (0.06)	1.07 (0.18)
Train	0.71 (0.09)	0.71 (0.08)	1.06 (0.15)
Subway	0.65 (0.06)	0.75 (0.19)	0.90 (0.14)

In every noise scenario, we run the SVR-MKC ten times and report the average results. For each independent run, 1000 observations are randomly extracted from the training set for training. Then, the classifier that yields the best performance on the development set is picked up from the grid search of the parameters. For the SVR-MKC, parameter C is searched from $2^{[1:1:5]}$. Parameter l is searched from $[0, 0.025n, 0.05n, 0.075n, 0.1n]$. Only one RBF kernel is used for each feature expression. The RBF kernel width σ_q is set to $0.5\gamma_q$, $q = 1, \ldots, Q$, where γ_q is the average Euclidean distance of the qth feature expression. At last, we report the performance of the selected classifier on the evaluation set. The parameter settings of the MK-SVM are the same as [98].

Fig. 3 gives an example of the ROC curve comparison of the SVR-MKC- and MK-SVM-based VADs in the car noise scenario. From the figure, the SVR-MKC can achieve a higher accuracy (80.14%) than the SVR-MMCs with the MO-MP window lengths of 2 (79.10%) and 14 (80.10%). However, its unfortunate to observe that there is still a clear gap between the unsupervised methods and their supervised counterparts. Moreover, the SVR-MKC does not yield a ROC curve that can cover the ROC curves of the SVR-MMCs, which is much different from the supervised MK-SVM. Therefore, the SVR-MKC still has an enough space for improvement.

Table VII lists the performance comparisons of the SVR-MKC- and MK-SVM-based VADs in the seven noise scenarios. From the table, we can see the following: 1) The SVR-MKC is not much worse than the MK-SVM (with perfect labeling) and can even substitute the MK-SVM in the `Street` noise scenario. 2) The SVR-MKC has better accuracies than the SVR-MMC in most of the noise scenarios.

Table VIII lists the CPU time on prediction. From the table, we can see that the prediction of a long utterance with an arrange of [450, 750] s long can be finished within 2 s, which fully meets the real-time demand of the VAD.

X. CONCLUSION

In this paper, we have proposed a SVR-based MMC algorithm. Specifically, we have first used the SVR as the core of the MMC problem and have relaxed the SVR-based nonconvex integer MMC problem as a convex optimization problem. Then, we have solved the relaxed problem iteratively by CPA. Moreover, we have decomposed each cutting-plane subproblem to a serial supervised SVR problem by a new GELM algorithm. At last, we have solved each SVR via the CPSP algorithm. For real-world applications, we have further extended the SVR-MMC algorithm to the MKC scenario and the multiclass clustering scenario. The SVR-MMC and its two extensions, i.e., SVR-MKC and SVR-M3C, have the following three advantages. The first one is that they are formulated as convex optimization problems, so that global optimum solutions are available. The second one is that they have theoretical linearithmic time complexities and empirically determined linear time complexities with both linear and nonlinear kernels. The third one is that, if they are used as unsupervised learning methods, their prediction time complexities are irrelevant to the training set size. The experimental results have shown that the proposed SVR-MMC, SVR-MKC, and SVR-M3C algorithms can achieve higher clustering accuracies than several state-of-the-art clustering methods and MMC algorithms. We have also applied the SVR-MKC to VAD. The experimental results have shown that the SVR-MKC-based VAD can combine the advantages of multiple acoustic features together, achieve good performances that are close to the supervised MK-SVM-based VAD, and meet the real-time demand of VAD.

Appendix A
Proof of Theorem 1

The key point of the proof is to prove that the loss functions of problems (8) and (7) are equivalent.

Given the current label vector $\mathbf{y} \in \mathcal{B}_1 \subseteq \mathbb{R}^n$, the empirical Laplacian loss of the SVR (7) is

$$\begin{aligned}\ell_l &= \frac{1}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*)\\ &= \frac{1}{n}\sum_{i=1}^{n}\big(\max\left(0, y_i - \mathbf{w}^T\phi(\mathbf{x}_i)\right)\\ &\qquad + \max(0, -y_i + \mathbf{w}^T\phi(\mathbf{x}_i))\big)\\ &= \frac{1}{n}\sum_{i=1}^{n}\max\left(y_i - \mathbf{w}^T\phi(\mathbf{x}_i), -y_i + \mathbf{w}^T\phi(\mathbf{x}_i)\right) \quad (54)\end{aligned}$$

which can be also rewritten as

$$\begin{aligned}\ell_l = \frac{1}{n}\sum_{i=1}^{n}\max_{c_i\in\{0,1\}}\big((1-c_i)\left(y_i - \mathbf{w}^T\phi(\mathbf{x}_i)\right)\\ + c_i\left(-y_i + \mathbf{w}^T\phi(\mathbf{x}_i)\right)\big). \quad (55)\end{aligned}$$

Letting $g_i = 1 - 2c_i$ leads to

$$\begin{aligned}\ell_l &= \frac{1}{n}\sum_{i=1}^{n}\max_{g_i\in\{-1,1\}}\left(g_i\left(y_i - \mathbf{w}^T\phi(\mathbf{x}_i)\right)\right)\\ &= \max_{\mathbf{g}\in\{-1,1\}^n}\left(\frac{1}{n}\sum_{i=1}^{n}g_i(y_i - \mathbf{w}^T\phi(\mathbf{x}_i)\right) \triangleq \xi. \quad (56)\end{aligned}$$

Theorem 1 is proved.

Appendix B
Proof of Theorem 2

According to the CPA theory, the most violated $\mathbf{y}$ is the one that would result in the largest value of $-E(\mathbf{y};\boldsymbol{\alpha})$ [in (12)] at the current solution point $(\boldsymbol{\mu},\boldsymbol{\alpha})$, which is just the following optimization problem:

$$\max_{\mathbf{y}\in\mathcal{B}_0} -E(\mathbf{y};\boldsymbol{\alpha}) = \max_{\mathbf{y}\in\mathcal{B}_0} -\boldsymbol{\alpha}^T\mathbf{G}^T\mathbf{y} + \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{G}^T\mathbf{K}\mathbf{G}\boldsymbol{\alpha}. \quad (57)$$

Because the second item is irrelevant to the optimization problem, (57) is equivalent to (16), which can be efficiently solved in the same way as [41, Proposition 2].

Appendix C
Proof of Theorem 3

First of all, we should note that, although the variable vector $\boldsymbol{\mu}$ in (15) only shows $|\Omega|$ elements, it is, in fact, a long sparse vector with only the first $|\Omega|$ elements (possibly) not being zeros and all other elements being zeros. That is to say, the complete definition of any $\mathcal{M}_k$, $k \in \mathbb{N}$, should be defined as

$$\mathcal{M}_k = \left\{\boldsymbol{\mu}\middle|\begin{cases}\sum_{t=1}^{k}\mu_t = 1, \mu_t \geq 0, & \text{if } t <= k\\ \mu_t = 0, & \text{otherwise}\end{cases}\right\}. \quad (58)$$

Thus, we can have

$$\mathcal{M}_1 \subset \mathcal{M}_2 \subset \ldots \subset \mathcal{M}_{|\Omega|-1} \subset \mathcal{M}_{|\Omega|} \subset \ldots \subset \mathcal{M}. \quad (59)$$

From (59), we could know that any ELM solution point $(\boldsymbol{\mu}_k^i, \boldsymbol{\alpha}_k^i) \in \mathcal{M}_k \times \mathcal{A}$ is also a point in $\mathcal{M} \times \mathcal{A}$, which means that any past ELM solution point can contribute to the construction of the cutting-plane model of the ELM algorithm for the current problem (15), since ELM algorithm also belongs to the CPA category. Therefore, if we denote the objective value of the kth cutting-plane subproblem as J_k, the following descendent relation is guaranteed:

$$J_1 \geq J_2 \geq \cdots \geq J_{|\Omega|-1} \geq J_{|\Omega|} \geq \cdots \geq J \quad (60)$$

where J is the global minimum value of the SVR-MMC.

However, if we do not inherit the historical ELM solution points, we have to construct a new cutting-plane model of the ELM algorithm in each new cutting-plane subproblem, so that the model of the ELM in $\mathcal{M}_{k-1} \times \mathcal{A}$ has no relation with that in $\mathcal{M}_k \times \mathcal{A}$, which means that $J_{k-1} \geq J_k$ is not guaranteed.

Appendix D
Proof of Theorem 4

The most violated $\mathbf{Y}$ is the one that would result in the largest value of $-E_\Sigma(\mathbf{Y};\boldsymbol{\alpha})$ [in (46)] at the current solution point $(\boldsymbol{\mu},\boldsymbol{\alpha})$, which is just the following optimization problem:

$$\begin{aligned}&\max_{\mathbf{Y}\in\mathcal{B}_2} -E_\Sigma(\mathbf{Y};\boldsymbol{\alpha})\\ &\quad = \max_{\mathbf{Y}\in\mathcal{B}_2} -\sum_{p=1}^{P}\boldsymbol{\alpha}_p^T\mathbf{G}_p^T\bar{\bar{\mathbf{y}}}_p + \frac{1}{2}\sum_{p=1}^{P}\boldsymbol{\alpha}_p^T\mathbf{G}_p^T\mathbf{K}\mathbf{G}_p\boldsymbol{\alpha}_p. \quad (61)\end{aligned}$$

Because the second item is irrelevant to the optimization problem, (61) is equivalent to (49).

Appendix E
Proof of Theorem 8

The solution of problem (49) should satisfy the following two important constraints:

1) Each row (sample) of $\mathbf{Y}$ can only have one "1." All other values of the row should be "$1/(P-1)$." That is to say, $\bar{\mathbf{y}} \in \mathcal{B}_{\bar{\mathbf{y}}}$.

2) Each column (class) of $\mathbf{Y}$ should satisfy the class balance constraint. That is to say, $-l/(P-1) \leq \bar{\bar{\mathbf{y}}}^T\mathbf{1} \leq l$.

First, it is obvious that output $\mathbf{Y}$ of Algorithm 6 satisfies the aforementioned two items. Then, we prove that $\mathbf{Y}$ is the optimal solution. According to Algorithm 6, we can obtain the sorted $\mathbf{c}$, denoted as $\mathbf{c}'$, and the aligned $\mathbf{Y}$, denoted as $\mathbf{z}'$. Suppose another $\mathbf{Y}^+$ differs from $\mathbf{Y}$ in just two labels, which are denoted as $\bar{y}_{i,p}$ and $\bar{y}_{j,q}$. If we order $\mathbf{Y}^+$ to the same vector as $\mathbf{z}'$, we can align $\mathbf{c}$ to another vector $\mathbf{c}^+$. $\mathbf{c}'$ and $\mathbf{c}^+$ differ from each other in just two positions that are related to labels $\bar{y}_{i,p}$ and $\bar{y}_{j,q}$. Suppose the two elements at the two positions of $\mathbf{c}'$ are c'_a and c'_b, where a and b are the two indexes of the positions. We have $c^+_a = c'_b$ and $c^+_b = c'_a$. According to the fact that $\bar{y}_{i,p} \neq \bar{y}_{j,q}$ (so as to z'_a and z'_b) and the procedure of Algorithm 6, we must have $z'_a = 1$ and $z'_b = -1/(P-1)$. Based on the fact that $c'_a < c'_b$, the following inequality holds:

$$z'_a c'_a + z'_b c'_b < z'_a c'_b + z'_b c'_a = z'_a c^+_a + z'_b c^+_b$$

which means that $\mathbf{Y}$ can achieve lower objective value than $\mathbf{Y}^+$.

It is obvious that the sorting of $\{\boldsymbol{\alpha}_p^T \mathbf{G}_p^T\}_{p=1}^P$ is the most time-consuming part of Algorithm 6, which has an average time complexity of $\mathcal{O}(Pn \log Pn)$. Theorem 8 is proved.

Acknowledgment

The authors would like to thank the editors and the anonymous referees for their valuable advice, which greatly improved the quality of this paper. The authors would also like to thank the researchers who opened the codes of their excellent works, which greatly alleviated our workload on constructing the baseline.

References

[1] P. Maji, "Fuzzy-rough supervised attribute clustering algorithm and classification of microarray data," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 1, pp. 222–233, Feb. 2011.
[2] D. Wang, T. Li, S. Zhu, and Y. Gong, "Ihelp: An intelligent online helpdesk system," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 1, pp. 173–182, Feb. 2011.
[3] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Stat. Prob.*, 1967, vol. 1, pp. 281–297.
[4] J. A. Hartigan and M. A. Wong, "A k-means clustering algorithm," *Appl. Stat.*, vol. 28, pp. 100–108, 1979.
[5] C. Ding and X. He, "K-means clustering via principal component analysis," in *Proc. 21st Int. Conf. Mach. Learn.*, 2004, pp. 225–232.
[6] G. J. McLachlan and D. Peel, *Finite Mixture Models*. Hoboken, NJ: Wiley-Interscience, 2000.
[7] X. L. Xie and G. Beni, "A validity measure for fuzzy clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 8, pp. 841–847, Aug. 1991.
[8] I. Gath and A. B. Geva, "Unsupervised optimal fuzzy clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 773–780, Jul. 1989.
[9] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.
[10] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 849–856.
[11] R. Kannan, S. Vempala, and A. Vetta, "On clusterings: Good, bad and spectral," *J. ACM*, vol. 51, no. 3, pp. 497–515, May 2004.
[12] W. Hu, W. Hu, N. Xie, and S. Maybank, "Unsupervised active learning based on hierarchical graph-theoretic clustering," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 5, pp. 1147–1161, Oct. 2009.
[13] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Trans. Inf. Theory*, vol. IT-21, no. 1, pp. 32–40, Jan. 1975.
[14] X. T. Yuan and S. Z. Li, "Stochastic gradient kernel density mode-seeking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2009, pp. 1926–1931.
[15] X. T. Yuan, B. G. Hu, and R. He, "Agglomerative mean-shift clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 2, pp. 209–219, Feb. 2012.
[16] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans, "Maximum margin clustering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, vol. 17, pp. 1537–1544.
[17] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995.
[18] A. Ben-Hur, D. Horn, H. Siegelmann, and V. Vapnik, "A support vector clustering method," in *Proc. 15th Int. Conf. Pattern Recog.*, 2000, vol. 2, pp. 724–727.
[19] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik, "Support vector clustering," *J. Mach. Learn. Res.*, vol. 2, pp. 125–137, Dec. 2001.
[20] W. Jeen-Shing and C. Jen-Chieh, "A cluster validity measure with outlier detection for support vector clustering," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 1, pp. 78–89, Feb. 2008.
[21] H. Valizadegan and R. Jin, "Generalized maximum margin clustering and unsupervised kernel learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, vol. 19, pp. 1417–1425.
[22] L. Xu and D. Schuurmans, "Unsupervised and semi-supervised multi-class support vector machines," in *Proc. 20th Nat. Conf. Artif. Intell.*, 2005, vol. 2, pp. 904–910.
[23] F. Gieseke, T. Pahikkala, and O. Kramer, "Fast evolutionary maximum margin clustering," in *Proc. 26th Int. Conf. Mach. Learn.*, 2009, pp. 361–368.
[24] K. Zhang, I. W. Tsang, and J. T. Kwok, "Maximum margin clustering made practical," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 1119–1126.
[25] K. Zhang, I. W. Tsang, and J. T. Kwok, "Maximum margin clustering made practical," *IEEE Trans. Neural Netw.*, vol. 20, no. 4, pp. 583–596, Apr. 2009.
[26] B. Zhao, F. Wang, and C. Zhang, "Efficient maximum margin clustering via cutting plane algorithm," in *Proc. 8th SIAM Int. Conf. Data Min.*, 2008, pp. 751–762.
[27] F. Wang, B. Zhao, and C. S. Zhang, "Linear time maximum margin clustering," *IEEE Trans. Neural Netw.*, vol. 21, no. 2, pp. 319–332, Feb. 2010.
[28] Y. Hu, J. Wang, N. Yu, and X. S. Hua, "Maximum margin clustering with pairwise constraints," in *Proc. 8th IEEE Int. Conf. Data Min.*, 2008, pp. 253–262.
[29] J. Wu and X. L. Zhang, "Sparse kernel maximum margin clustering," *Neural Netw. World*, vol. 21, no. 6, pp. 551–574, 2011.
[30] H. Zeng and Y. M. Cheung, "Semi-supervised maximum margin clustering with pairwise constraints," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 5, pp. 926–939, May 2012.
[31] A. L. Yuille and A. Rangarajan, "The concave–convex procedure," *Neural Comput.*, vol. 15, no. 4, pp. 915–936, Apr. 2003.
[32] A. J. Smola, S. V. N. Vishwanathan, and T. Hofmann, "Kernel methods for missing variables," in *Proc. 10th Int. Workshop Artif. Intell. Stat.*, 2005, pp. 325–332.
[33] J. E. Kelley, "The cutting-plane method for solving convex programs," *J. Soc. Ind. Appl. Math.*, vol. 8, no. 4, pp. 703–712, Dec. 1960.
[34] T. Joachims, "Training linear SVMs in linear time," in *Proc. 12th ACM Int. Conf. Knowl. Disc. Data Min.*, 2006, pp. 217–226.
[35] B. Schölkopf, A. Smola, and K. R. Müller, "Kernel principal component analysis," in *Proc. Artif. Neural Netw.*, 1997, pp. 583–588.
[36] R. He, B. G. Hu, W. S. Zheng, and X. Kong, "Robust principal component analysis based on maximum correntropy criterion," *IEEE Trans. Image Process.*, vol. 20, no. 6, pp. 1485–1494, Jun. 2011.
[37] O. Chapelle and A. Zien, "Semi-supervised classification by low density separation," in *Proc. 10th Int. Workshop Artif. Intell. Stat.*, 2005, pp. 1–8.
[38] B. Schölkopf and A. J. Smola, *Learning With Kernels*. Cambridge, MA: MIT Press, 2002.
[39] L. N. Trefethen and D. Bau, *Numerical Linear Algebra*. Philadelphia, PA: SIAM, 1997.
[40] B. Zhao, J. T. Kwok, and C. S. Zhang, "Multiple kernel clustering," in *Proc. 9th SIAM Int. Conf. Data Min.*, 2009, pp. 638–649.

[41] Y. F. Li, I. W. Tsang, J. T. Kwok, and Z. H. Zhou, "Tighter and convex maximum margin clustering," in *Proc. 12th Int. Conf. Artif. Intell. Stat.*, Clearwater Beach, FL, 2009, pp. 344–351.

[42] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[43] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet, "SimpleMKL," *J. Mach. Learn. Res.*, vol. 9, pp. 2491–2521, Nov. 2008.

[44] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Stat. Comput.*, vol. 14, no. 3, pp. 199–222, Aug. 2004.

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

[46] C. H. Teo, A. Smola, S. V. N. Vishwanathan, and Q. V. Le, "A scalable modular convex solver for regularized risk minimization," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2007, pp. 727–736.

[47] V. Franc and S. Sonnenburg, "Optimized cutting plane algorithm for support vector machines," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 320–327.

[48] C. J. C. Burges, "Simplified support vector decision rules," in *Proc. Int. Conf. Mach. Learn.*, 1996, pp. 71–77.

[49] B. Zhao, F. Wang, and C. Zhang, "Efficient multiclass maximum margin clustering," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 1248–1255.

[50] T. Joachims, "A support vector method for multivariate performance measures," in *Proc. 22nd Int. Conf. Mach. Learn.*, 2005, pp. 377–384.

[51] T. Joachims, T. Finley, and C. N. J. Yu, "Cutting-plane training of structural SVMs," *Mach. Learn.*, vol. 77, no. 1, pp. 27–59, Oct. 2009.

[52] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *J. Mach. Learn. Res.*, vol. 5, pp. 27–72, Dec. 2004.

[53] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the SMO algorithm," in *Proc. 21th Int. Conf. Mach. Learn.*, 2004, pp. 6–13.

[54] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf, "Large scale multiple kernel learning," *J. Mach. Learn. Res.*, vol. 7, pp. 1531–1565, Dec. 2006.

[55] A. Zien and C. S. Ong, "Multiclass multiple kernel learning," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 1191–1198.

[56] Z. Xu, R. Jin, I. King, and M. R. Lyu, "An extended level method for efficient multiple kernel learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, vol. 21, pp. 1825–1832.

[57] H. Yang, Z. Xu, J. Ye, I. King, and M. R. Lyu, "Efficient sparse generalized multiple kernel learning," *IEEE Trans. Neural Netw.*, vol. 22, no. 3, pp. 433–446, Mar. 2011.

[58] M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K. R. Müller, and A. Zien, "Efficient and accurate LP-norm multiple kernel learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, vol. 22, pp. 997–1005.

[59] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet, "More efficiency in multiple kernel learning," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 775–782.

[60] C. Lemaréchal, A. Nemirovskii, and Y. Nesterov, "New variants of bundle methods," *Math. Program.*, vol. 69, no. 1, pp. 111–147, Jul. 1995.

[61] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *J. Aritif. Intell. Res.*, vol. 2, no. 1, pp. 263–286, Aug. 1994.

[62] K. Cherkauer, "Human expert-level performance on a scientific image analysis task by a system using combined artificial neural networks," in *Proc. Working Notes AAAI Workshop Integr. Multiple Learned Models*, 1996, pp. 15–21.

[63] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[64] G. Martinez-Muoz, D. Hernández-Lobato, and A. Suárez, "An analysis of ensemble pruning techniques based on ordered aggregation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 245–259, Feb. 2009.

[65] L. Chen and M. Kamel, "A generalized adaptive ensemble generation and aggregation approach for multiple classifier systems," *Pattern Recognit.*, vol. 42, no. 5, pp. 629–644, May 2009.

[66] L. Nanni and A. Lumini, "Fuzzy bagging: A novel ensemble of classifiers," *Pattern Recognit.*, vol. 39, no. 3, pp. 488–490, Mar. 2006.

[67] C. W. Hsu and C. J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Trans. Neural Netw.*, vol. 13, no. 2, pp. 415–425, Mar. 2002.

[68] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: A unifying approach for margin classifiers," *J. Mach. Learn. Res.*, vol. 1, pp. 113–141, Sep. 2001.

[69] K. Crammer and Y. Singer, "On the learnability and design of output codes for multiclass problems," *Mach. Learn.*, vol. 47, no. 2/3, pp. 201–233, May/Jun. 2002.

[70] O. Pujol, P. Radeva, and J. Vitria, "Discriminant ECOC: A heuristic method for application dependent design of error correcting output codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 6, pp. 1007–1012, Jun. 2006.

[71] O. Pujol, S. Escalera, and P. Radeva, "An incremental node embedding technique for error correcting output codes," *Pattern Recogn.*, vol. 41, no. 2, pp. 713–725, Feb. 2008.

[72] S. Escalera, D. M. J. Tax, O. Pujol, P. Radeva, and R. P. W. Duin, "Subclass problem-dependent design for error-correcting output codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 6, pp. 1041–1054, Jun. 2008.

[73] S. Escalera, O. Pujol, and P. Radeva, "On the decoding process in ternary error-correcting output codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 120–134, Jan. 2010.

[74] G. Fung and O. Mangasarian, "Multicategory proximal support vector machine classifiers," *Mach. Learn.*, vol. 59, no. 1/2, pp. 77–97, May 2005.

[75] J. Weston and C. Watkins, "Support vector machines for multi-class pattern recognition," in *Proc. 7th Eur. Symp. Artif. Neural Netw.*, 1999, vol. 4, no. 6, pp. 219–224.

[76] Y. Guermeur, "Combining discriminant models with new multi-class SVMs," *Pattern Anal. Appl.*, vol. 5, no. 2, pp. 168–179, 2002.

[77] Y. Lee, Y. Lin, and G. Wahba, "Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data," *J. Amer. Stat. Assoc.*, vol. 99, no. 465, pp. 67–81, Jan. 2004.

[78] Y. Zhang and Z. H. Zhou, "Cost-sensitive face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 10, pp. 1758–1769, Oct. 2010.

[79] P. Chen, K. Y. Lee, T. J. Lee, Y. J. Lee, and S. Y. Huang, "Multiclass support vector classification via coding and regression," *Neurocomputing*, vol. 73, no. 7–9, pp. 1501–1512, Mar. 2010.

[80] S. Ghorai, A. Mukherjee, and P. K. Dutta, "Discriminant analysis for fast multiclass data classification through regularized kernel function approximation," *IEEE Trans. Neural Netw.*, vol. 21, no. 6, pp. 1020–1029, Jun. 2010.

[81] R. Collobert and S. Bengio, "SVMTorch: Support vector machines for large-scale regression problems," *J. Mach. Learn. Res.*, vol. 1, pp. 143–160, Sep. 2001.

[82] W. F. Zhang, D. Q. Dai, and H. Yan, "Framelet kernels with applications to support vector regression and regularization networks," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 4, pp. 1128–1144, Aug. 2010.

[83] S. Kim and S. Boyd, "A minimax theorem with applications to machine learning, signal processing, and finance," in *Proc. 46th IEEE Int. Conf. Decision, Control*, 2007, pp. 751–758.

[84] T. Joachims and C. N. J. Yu, "Sparse kernel SVMs via cutting-plane training," *Mach. Learn.*, vol. 76, no. 2/3, pp. 179–193, Sep. 2009.

[85] P. C. Chen, T. J. Lee, Y. J. Lee, and S. Y. Huang, Multiclass Support Vector Classification via Regression. [Online]. Available: http://www.stat.sinica.edu.tw/syhuang/papersdownload/MC-reg-121006.pdf

[86] A. Benyassine, E. Shlomot, H. Y. Su, D. Massaloux, C. Lamblin, and J. P. Petit, "ITU-T Recommendation G. 729 Annex B: A silence compression scheme for use with G. 729 optimized for V. 70 digital simultaneous voice and data applications," *IEEE Commun. Mag.*, vol. 35, no. 9, pp. 64–73, Sep. 1997.

[87] Speech Processing, Transmission and Quality Aspects (STQ); Distributed Speech Recognition; Advanced Front-End Feature Extraction Algorithm; Compression Algorithms, ETSI ES 202 050, 2007.

[88] C. W. Hsu, C. C. Chang, and C. J. Lin, A practical guide to support vector classification, Dept. Comput. Sci., Nat. Taiwan Univ., Taipei, Taiwan. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

[89] D. Enqing, L. Guizhong, Z. Yatong, and Z. Xiaodi, "Applying support vector machines to voice activity detection," in *Proc. Int. Conf. Signal Process.*, 2002, vol. 2, pp. 1124–1127.

[90] J. Ramírez, P. Yélamos, J. M. Górriz, and J. C. Segura, "SVM-based speech endpoint detection using contextual speech features," *Electron. Lett.*, vol. 42, no. 7, pp. 426–428, Mar. 2006.

[91] D. Kim, K. W. Jang, and J. Chang, "A new statistical voice activity detection based on UMP test," *IEEE Signal Process. Lett.*, vol. 14, no. 11, pp. 891–894, Nov. 2007.

[92] S. I. Kang, Q. H. Jo, and J. H. Chang, "Discriminative weight training for a statistical model-based voice activity detection," *IEEE Signal Process. Lett.*, vol. 15, pp. 170–173, 2008.

[93] Q. H. Jo, J. H. Chang, J. W. Shin, and N. S. Kim, "Statistical model-based voice activity detection using support vector machine," *IET Signal Process.*, vol. 3, no. 3, pp. 205–210, May 2009.

[94] J. W. Shin, J. H. Chang, and N. S. Kim, "Voice activity detection based on statistical models and machine learning approaches," *Comput. Speech Lang.*, vol. 24, no. 3, pp. 515–530, Jul. 2010.

[95] T. Yu and J. H. L. Hansen, "Discriminative training for multiple observation likelihood ratio based voice activity detection," *IEEE Signal Process. Lett.*, vol. 17, no. 11, pp. 897–900, Nov. 2010.

[96] J. Wu and X. L. Zhang, "Maximum margin clustering based statistical VAD with multiple observation compound feature," *IEEE Signal Process. Lett.*, vol. 18, no. 5, pp. 283–286, May 2011.

[97] J. Wu and X. L. Zhang, "An efficient voice activity detection algorithm by combining statistical model and energy detection," *EURASIP J. Adv. Signal Process.*, vol. 2011, no. 1, pp. 18–27, Dec. 2011.

[98] J. Wu and X. L. Zhang, "Efficient multiple kernel support vector machine based voice activity detection," *IEEE Signal Process. Lett.*, vol. 18, no. 8, pp. 466–469, Aug. 2011.

[99] D. Cournapeau, S. Watanabe, A. Nakamura, and T. Kawahara, "Online unsupervised classification with model comparison in the variational Bayes framework for voice activity detection," *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 6, pp. 1071–1083, Dec. 2010.

[100] D. Ying, Y. Yan, J. Dang, and F. Soong, "Voice activity detection based on an unsupervised learning framework," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 8, pp. 2624–2633, Nov. 2011.

[101] D. Pearce and H. Hirsch, "The AURORA experimental framework for the performance evaluation of speech recognition systems under noisy conditions," in *Proc. ICSLP*, 2000, vol. 4, pp. 29–32.

[102] S. M. Kay, *Fundamentals of Statistical Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

Xiao-Lei Zhang (S'08) received the B.S. degree in education technology from Nanjing University of Posts and Telecommunications, Nanjing, China, in 2005 and the M.S. degree in signal and information processing from Nanjing University, Nanjing, China, in 2008. He is currently working toward the Ph.D. degree in information and communication engineering at Tsinghua University, Beijing, China.

His current research interests include the topics on machine learning, audio signal processing, and audio information retrieval. He has published over ten journal articles and conference papers.

Mr. Zhang was a recipient of several awards including the Major Award of Tsinghua University.

Ji Wu (M'06) received the B.S. and Ph.D. degrees from Tsinghua University, Beijing, China, in 1996 and 2001, respectively.

He is currently an Associate Professor and the Deputy Director of the Department of Electronic Engineering, Tsinghua University. His research interests include speech recognition, multimedia signal processing, pattern recognition, and machine learning.