Pattern Recognition 69 (2017) 286–298

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

FRSVC: Towards making support vector clustering consume less

Yuan Ping[a,b,*], Yingjie Tian[c], Chun Guo[b,d], Baocang Wang[a], Yuehua Yang[a]

[a] School of Information Engineering, Xuchang University, Xuchang 461000, China
[b] Guizhou Provincial Key Laboratory of Public Big Data, Guiyang 550025, China
[c] CAS Research Center on Fictitious Economy & Data Science, Beijing 100190, China
[d] College of Computer Science & Technology, GuiZhou University, GuiYang, 550025, China

ARTICLE INFO

Article history:
Received 31 July 2016
Revised 12 February 2017
Accepted 26 April 2017
Available online 27 April 2017

Keywords:
Large-scale data
Support vector clustering
Dual coordinate descent
Sampling strategy

ABSTRACT

In spite of with great advantage of discovering arbitrary shapes of clusters, support vector clustering (SVC) is frustrated by large-scale data, especially on resource limited platform. It is due to pricey storage and computation consumptions from solving dual problem and labeling clusters upon the pre-computed kernel matrix and sampling point pairs, respectively. Towards on it, we first present a dual coordinate descent method to reformulate the solver that leads to a flexible training phase carried out on any runtime platform with/without sufficient memory. Then, a novel labeling phase who does connectivity analysis between two nearest neighboring decomposed convex hulls referring to clusters is proposed, in which a new designed strategy namely sample once connected checking first tries to reduces the scope of sampling analysis. By integrating them together, a faster and reformulated SVC (FRSVC) is created with less consumption achieved according to comparative analysis of time and space complexities. Furthermore, experimental results confirm a significant improvement on flexibility of selective efficiency without losing accuracy, with which a balance can be easily reached on the basis of resources a platform equipped.

1. Introduction

Clustering focuses on forming natural groupings of data samples that maximize intra-cluster similarity and minimize inter-cluster similarity. Up to now, inspired by support vector machines (SVMs), support vector clustering (SVC) has attracted many researchers for its advantages of handling clusters with arbitrary shape [1,2]. A wide variety of domains can be found related to it, e.g., information retrieval, instance-based learning, pattern decisioning, and traffic identification etc. [1]. However, building a good clustering model requires a large number of valid training samples and a hard iterative analysis under some metrics; hence, it is frequently not possible for individuals with resource limited equipment to build their own model for those easily accumulated large-scale data. Actually, two critical phases of SVC, i.e., training phase to estimate a support function by solving the dual problem and cluster labeling phase, have been always considered as the major bottlenecks [1–5]. Therefore, a viable solution for this problem is to make SVC consume less, either computation or storage requirements or both, without losing performance on accuracy.

For a data set $\mathcal{X}$ with N data samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, where $\mathbf{x}_i \in \mathbb{R}^d (i \in [1, N])$ in data space, the two pricey phases have being widely investigated. As a quadratic programming problem, the training phase usually takes runtime time from $O(N^2)$ to $O(N^3)$ which depends on the actual case [3,6,7]. In practical use, an uncertain number of iterations for the final coefficient vector $\boldsymbol{\beta}$ and the selection of working set make it consuming much more. Meanwhile, a space complexity of $O(N^2)$ is generally required. By contrast, the time complexity of labeling phase is $O(mN^2)$ with m suggested to be in [10, 20] is the sample rate on each edge connecting samples pair. Frequently, N is shared by both of the two phases. In the literature, therefore, many insightful works [2–5,8,9] have been done towards on reducing the number of participated data samples N, some [3,4] of them also consider the reduction of m. However, they reduce N to optimize runtime not space requirement, and the control of iterations is not taken into account; they reduce m in average but it is far from the ultimate 1. Furthermore, we additionally expect to cut down the exponent in runtime of training phase.

With these considerations, in this paper, we propose a faster and reformulated SVC (FRSVC) to maximize the capability of doing data clustering even on a low-end with limited resources. To achieve less consumption, FRSVC firstly tries to control the iterations on demand while keeps the quality of $\boldsymbol{\beta}$. Also in the training phase, it supports utilizing either the pre-computed kernel matrix

* Corresponding author.
E-mail address: pingyuan@bupt.edu.cn (Y. Ping).

http://dx.doi.org/10.1016/j.patcog.2017.04.025
0031-3203/

or not, for further benefit of efficiency or storage, respectively. Obviously, the balance is achieved with respect to the trade-off between computation and storage which a platform affords. Then, in the labeling phase, the ultimate expectation of one time for the average sample rate is obtained by FRSVC to make connectivity decision of two neighboring components. The main contributions lie in:

- The proposal of a reformative solver for model training which employs a dual coordinate descent method (DCD) to reformulate the procedure of solving dual problem. By altering the original clustering problem to a classification issue, it achieves the expected result by a linear method. Not only the computation is simplified, but also those unnecessary storage consumptions can be effectively avoided.
- In the labeling phase, to reduce the scope of sampling analysis for further improvement on efficiency, a novel labeling method is designed. To get the status of connectivity between two nearest neighboring clusters, it first finds two types of prototype for clusters, i.e., convex hull and stable equilibrium vector (SEV). Then the sampling analysis is done between one SV and one SEV, respectively, from two nearest neighboring convex hulls. With a newly designed strategy of connection checking first, to reach a comparable accuracy, once sample ($m\downarrow$) becomes the maximum sample rate in each round that makes it achieving much less consumption.

The remainder of this paper is organized as follows: In Section 2, we briefly describe the classic phases involved in SVC. In Section 3, we first reformulate the classic SVC in terms of an equivalent form whose dual problem can be solved by DCD method effectively, and the corresponding solver is proposed. Meanwhile, we present novel strategies of cluster labeling. Then the basic framework of the proposed FRSVC is shown. Theoretical analysis and experimental results are done in Section 4. And Section 5 gives our review of those related works. Finally, conclusions are drawn in the last section, as well as the future works to be investigated.

2. Preliminaries of SVC

2.1. Estimation of a trained support function

For the data set $\mathcal{X}$, its data samples can be mapped to a high-dimensional feature space from data space through a nonlinear function $\Phi(\cdot)$. Then, SVC tries to find a sphere with the minimal radius which contains most of the mapped data samples. This sphere, when mapped back to the data space, can be partitioned into several components, each one encloses a isolated cluster of samples. In mathematical formulation, the spherical radius R subjects to

$$\begin{aligned} &\min_{R,\boldsymbol{\alpha},\xi_i} \quad R^2 + C\sum_i \xi_i \\ &\text{s.t.} \quad ||\Phi(\mathbf{x}_i) - \boldsymbol{\alpha}||^2 \le R^2 + \xi_i, \end{aligned} \tag{1}$$

where $\boldsymbol{\alpha}$ is the center of the sphere, ξ_i is a slack variable, and C is a constant controlling the penalty of noise. Following Refs. [2,5,10], the expected sphere is estimated by a support function which is defined as a positive scalar function $f : \mathbb{R}^n \to \mathbb{R}^+$. Since the support function is actually constructed by SVs, we can estimate it by SVs extracted by solving a dual problem in Eq. (2) where $\mathbf{x}_i$ corresponds to coefficient $\beta_i (i = 1, \ldots, N)$ if its $0 < \beta_i < C$ is a SV.

$$\begin{aligned} &\max_{\beta_j} \sum_j K(\mathbf{x}_j, \mathbf{x}_j)\beta_j - \sum_{i,j} \beta_i\beta_j K(\mathbf{x}_i, \mathbf{x}_j) \\ &\text{s.t.} \quad \sum_j \beta_j = 1, \quad 0 \le \beta_j \le C, \quad j = 1, \ldots, N. \end{aligned} \tag{2}$$

By optimizing Eq. (2) with Gaussian kernel $K(\mathbf{x}_i, \mathbf{x}_j) = e^{-q||\mathbf{x}_i - \mathbf{x}_j||^2}$, the objective trained support function can be formulated by the squared radial distance of the image of $\mathbf{x}$ from the sphere center $\boldsymbol{\alpha}$ given by

$$\begin{aligned} f(\mathbf{x}) = ||\Phi(\mathbf{x}) - \boldsymbol{\alpha}||^2 &= K(\mathbf{x}, \mathbf{x}) - 2\sum_j \beta_j K(\mathbf{x}_j, \mathbf{x}) \\ &+ \sum_{i,j} \beta_i\beta_j K(\mathbf{x}_i, \mathbf{x}_j). \end{aligned} \tag{3}$$

$$\boldsymbol{\alpha} = \sum_j \beta_j \Phi(\mathbf{x}_j). \tag{4}$$

Theoretically, the radius R of the hypersphere is usually defined by the square root of $f(\mathbf{x}_i)$ where $\mathbf{x}_i$ is one of SVs.

2.2. Cluster assignments

Since SVs locate on the border of clusters, a simple graphical connected-component method can be used for cluster labeling. For any two samples, $\mathbf{x}_i$ and $\mathbf{x}_j$, we check m segmers on the line segment connecting them by traveling their images in the hypersphere. According to Eq. (3), $\mathbf{x}_i$ and $\mathbf{x}_j$ should be labeled the same cluster index while all the m segmers are always lying in the hypersphere, i.e., $f(\mathbf{x}_{\tilde{m}}) \le R^2$ for $\tilde{m} \in [1, m]$. Otherwise, they will be in two different clusters.

3. Framework of the proposed FRSVC

As described in Section 1, pricey time complexity and space complexity exist in the traditional solutions of SVC, in the literature. In this section, we first present methods for the training phase and the labeling phase, respectively. These methods are specially designed towards the root causes of the two phases' performance and flexibility. Then, we give the description of the basic frameworks of FRSVC.

3.1. Dual problem solver by dual coordinate descent

Solving the dual problem (2) is the crucial bottleneck for the training phase. Because the kernel matrix with elements $K(\mathbf{x}_i, \mathbf{x}_j)(i, j \in [1, N])$ is expected to be pre-computed before iterative optimization. Its computation and memory consumption might exceed the capacity that a general platform can supply, especially for a large N. In that case, the traditional solutions prefer mathematical method on matrix blocks partitioned from the kernel matrix. However, without considering the physical meaning of the matrix blocks, they are not only frequently with high time complexity, but also not suitable for parallelization with a high accuracy requirement. Therefore, in this section, we describe our DCD solver for the dual problem (2).

DCD is firstly used by Ref. [11] to solve large-scale linear SVM. It is efficient for reaching an $\epsilon-$accurate solution in $O(\log(1/\epsilon))$ iterations. Although SVC is a variant of one-class SVM, it can not be directly solved by DCD because of its nonlinear model. Thence, we need to reformulate it by a linear model.

Apparently, the dual problem of SVC in Eq. (2) can be described by

$$\begin{aligned} &\min_{\beta_j} \quad \sum_{i,j} \beta_i\beta_j K(\mathbf{x}_i, \mathbf{x}_j) - \sum_j K(\mathbf{x}_j, \mathbf{x}_j)\beta_j \\ &\text{s.t.} \quad \sum_j \beta_j = 1, \quad 0 \le \beta_j \le C, \quad j = 1, \ldots, N \end{aligned} \tag{5}$$

Let Q is the kernel matrix with element $Q_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)(i, j \in [1, N])$, $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots, \beta_N]^T$, and $\mathbf{e} = [1, 1, \ldots, 1]^T$. Since

$K(\mathbf{x}_i, \mathbf{x}_j) = 1$ for $\mathbf{x}_i = \mathbf{x}_j$, then Eq. (5) can be further reformulated by

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & \boldsymbol{\beta}^T Q \boldsymbol{\beta} - \boldsymbol{e}^T \boldsymbol{\beta} \\ \text{s.t.} \quad & \sum_j \beta_j = 1, \quad 0 \le \beta_j \le C, \quad j = 1, \ldots, N. \end{aligned} \tag{6}$$

Let $\tilde{Q}$ is a variant of kernel matrix Q, and its element $\tilde{Q}_{ij} = 2 \times Q_{ij} = 2 \times K(\mathbf{x}_i, \mathbf{x}_j)(i, j \in [1, N])$, then we have

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & \frac{1}{2} \boldsymbol{\beta}^T \tilde{Q} \boldsymbol{\beta} - \boldsymbol{e}^T \boldsymbol{\beta} \\ \text{s.t.} \quad & \sum_j \beta_j = 1, \quad 0 \le \beta_j \le C, \quad j = 1, \ldots, N. \end{aligned} \tag{7}$$

Eq. (7) is an obvious convex optimization problem, thus, it can always reach the globally optimal solution in region of constraint $D_{\text{svc}} = \{\boldsymbol{\beta} | \sum_j \beta_j = 1, 0 \le \beta_j \le C, j = 1, \ldots, N\}$. Similar with Refs. [8–10,12], we can also find an equivalent globally optimal solution if we relax D_{svc} by $D'_{\text{svc}} = \{\boldsymbol{\beta} | 0 \le \beta_j \le C, j = 1, \ldots, N\}$, where the strict condition of $\sum_j \beta_j = 1$ is removed. So, Eq. (2) can be reformulated by a similar form of linear SVM, i.e.,

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & \frac{1}{2} \boldsymbol{\beta}^T \tilde{Q} \boldsymbol{\beta} - \boldsymbol{e}^T \boldsymbol{\beta} \\ \text{s.t.} \quad & 0 \le \beta_j \le C, \quad j = 1, \ldots, N, \end{aligned} \tag{8}$$

where the expected diagonal matrix part D in $\tilde{Q}$ is removed or considered as $\mathbf{0}$ (i.e., $D_{ii} = 0$) for simplicity.

Although Eq. (8) has a similar form with the typical SVM, it can not be directly solved by DCD because, essentially, it is an unsupervised model. No label is supplied for any data sample when it is used to construct the corresponding solution. Therefore, we need to extend $\mathcal{X}$ to $\mathcal{X}'$ whose instance-label pairs are $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)\}$. Here y_i is fixed to $+1$ or -1 for $i = 1, \ldots, N$. Derived from Ref. [11], we describe our DCD solver for SVC in the following part of this section.

The optimization process of Eq. (8) starts from an initial point $\boldsymbol{\beta}^0 \in R^N$ and generates a sequences of vectors $\{\boldsymbol{\beta}^k\}_{k=1}^{\infty}$. We refer to the process from $\boldsymbol{\beta}^k$ to $\boldsymbol{\beta}^{k+1}$ as an outer iteration. In each outer iteration we have N inner iterations, so that sequentially $\beta_1, \beta_2, \ldots, \beta_N$ are updated. Each outer iteration thus generates vectors $\boldsymbol{\beta}^{k,i} \in R^N, i = 1, 2, \ldots, N+1$, such that $\boldsymbol{\beta}^{k,1} = \boldsymbol{\beta}^k$, $\boldsymbol{\beta}^{k,N+1} = \boldsymbol{\beta}^{k+1}$, and $\boldsymbol{\beta}^{k,i} = [\beta_1^{k+1}, \ldots, \beta_{i-1}^{k+1}, \beta_i^k, \ldots, \beta_N^k]^T$ for $\forall i = 2, \ldots, N$. To update $\boldsymbol{\beta}^{k,i}$ to $\boldsymbol{\beta}^{k,i+1}$, we fix the other variables and then solve the following one-variable sub-problem:

$$\begin{aligned} \min_{d} \quad & f(\boldsymbol{\beta}^{k,i} + \theta \boldsymbol{e}_i) \\ \text{s.t.} \quad & 0 \le \beta_i^k + \theta \le C, \end{aligned} \tag{9}$$

where $\boldsymbol{e}_i = [0, \ldots, 0, 1, 0, \ldots, 0]^T$. Then, the objective function of Eq. (9) is simple quadratic function of θ:

$$f(\boldsymbol{\beta}^{k,i} + \theta \boldsymbol{e}_i) = \frac{1}{2} \tilde{Q}_{ii} \theta^2 + \nabla_i f(\boldsymbol{\beta}^{k,i}) \theta + \text{constant}, \tag{10}$$

where $\nabla_i f$ is the ith component of the gradient ∇f. Apparently, Eq. (9) has an optimum at $\theta = 0$ if and only if $\nabla_i^P f(\boldsymbol{\beta}^{k,i}) = 0$, where $\nabla_i^P f(\boldsymbol{\beta})$ means the projected gradient

$$\nabla_i^P f(\boldsymbol{\beta}) = \begin{cases} \nabla_i f(\boldsymbol{\beta}) & \text{if } 0 < \beta_i < C \\ \min(\nabla_i f(\boldsymbol{\beta}), 0) & \text{if } \beta_i = 0 \\ \max(\nabla_i f(\boldsymbol{\beta}), 0) & \text{if } \beta_i = C. \end{cases} \tag{11}$$

If $\nabla_i^P f(\boldsymbol{\beta}^{k,i}) = 0$, we move to the index $i+1$ without updating $\beta_i^{k,i}$. Otherwise, we must find the optimal solution of Eq. (9). If $\tilde{Q}_{ii} > 0$, the solution is

$$\beta_i^{k,i} = \min\left(\max\left(\beta_i^{k,i} - \frac{\nabla_i f(\boldsymbol{\beta}^{k,i})}{\tilde{Q}_{ii}}, 0\right), C\right), \tag{12}$$

and then continue finding in the current index without moving to $i+1$.

Up to now, we can easily find that calculating $\tilde{Q}_{ii}$ and $\nabla_i f(\boldsymbol{\beta}^{k,i})$ are critical works for the optimal solution. First, according to the definition, $\tilde{Q}_{ii}$ can be calculated by

$$\tilde{Q}_{ii} = 2 \times K(\mathbf{x}_i, \mathbf{x}_i) + D_{ii} = 2, \tag{13}$$

i.e., it is a constant without being updated along with the iteration. Second, to evaluate $\nabla_i f(\boldsymbol{\beta}^{k,i})$, although we have

$$\nabla_i f(\boldsymbol{\beta}) = (\tilde{Q}\boldsymbol{\beta})_i - 1 = \sum_{j=1}^{N} \tilde{Q}_{ij} \beta_j - 1, \tag{14}$$

but we don't want to precompute them because of pricey storage consumption. Fortunately, as a variant of one-class SVM, we can, in feature space, define a decision function for SVC by its boundary, i.e., $y = \boldsymbol{w}^T \Phi(\mathbf{x}) + b$. Then, we get

$$\boldsymbol{w} = \sum_{i=1}^{N} \beta_i y_i \Phi(\mathbf{x}_i) = \sum_{i=1}^{N} \beta_i \Phi(\mathbf{x}_i). \tag{15}$$

So, by introducing the definition of $\tilde{Q}$, Eq. (14) becomes

$$\begin{aligned} \nabla_i f(\boldsymbol{\beta}) &= y_i \boldsymbol{w}^T \Phi(\mathbf{x}_i) - 1 + D_{ii} \beta_i \\ &= \boldsymbol{w}^T \Phi(\mathbf{x}_i) - 1. \end{aligned} \tag{16}$$

To apply Eq. (16), $\boldsymbol{w}$ should be maintained throughout the coordinate descent procedure. Thus, along with the update of β_i, we can maintain $\boldsymbol{w}$ by

$$\begin{aligned} \boldsymbol{w} &\leftarrow \boldsymbol{w} + (\beta_i - \hat{\beta}_i) y_i \Phi(\mathbf{x}_i) \\ &= \boldsymbol{w} + (\beta_i - \hat{\beta}_i) \Phi(\mathbf{x}_i), \end{aligned} \tag{17}$$

where $\hat{\beta}_i$ is the temporary coefficient value obtained in the previous iteration. Unfortunately, we cannot use an exactly nonlinear function $\Phi(\cdot)$ to map data sample $\mathbf{x}_i$ from the input space to the feature space. So we take Eq. (17) into Eq. (16), alternatively, to achieve an one-time update, i.e.,

$$\begin{aligned} \nabla_i f(\boldsymbol{\beta}^{k,i}) &\leftarrow [\boldsymbol{w} + (\beta - \hat{\beta}_i) \Phi(\mathbf{x}_i)]^T \Phi(\mathbf{x}_i) - 1 \\ &= \boldsymbol{w}^T \Phi(\mathbf{x}_i) + (\beta - \hat{\beta}_i) \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_i) - 1 \\ &= \underbrace{\sum_j \hat{\beta}_j K(\mathbf{x}_j, \mathbf{x}_i)}_{(k-1) \text{ iter}} + (\beta - \hat{\beta}_i) - 1. \end{aligned} \tag{18}$$

Therefore, to update $\nabla_i f(\boldsymbol{\beta})$ for finding the optimal solution, the current iterative value strongly depends on the weighted summation of the ith row of $\tilde{Q}$ (i.e., $\tilde{Q}_{i\cdot}$) in the previous iteration and the updated β_i.

Briefly, to solve the dual problem (8), our algorithm uses Eq. (18) to compute $\nabla_i f(\boldsymbol{\beta}^{k,i})$, checks the optimality of the sub-problem (9) by $\nabla_i^P f(\boldsymbol{\beta}^{k,i}) \stackrel{?}{=} 0$, updates β_i by Eq. (12). The procedure is detailed in Algorithm 1. The cost per iteration from $\boldsymbol{\beta}^k$ to $\boldsymbol{\beta}^{k+1}$ is $O(Nd)$. In addition, the memory requirement is flexible. With sufficient memory, one can continue to use $O(N^2)$ to store the full kernel matrix, and use search on demand to finish the calculation of line 5 in Algorithm 1. If the storage resource is very limited, we can either store $\mathbf{x}_1, \ldots, \mathbf{x}_N$ or split them into blocks (space complexity will be reduced to less than or equal $O(N)$), and then calculate the required kernel function values in sequence on demand for the outer iterations. If necessary, these computations related to kernel function can be easily migrated to distributed computational nodes.

Algorithm 1 DCD solver for dual problem (8) of SVC.

Require: Dataset $\mathcal{X}$, kernel width q, and penalty C
Ensure: Coefficient vector $\boldsymbol{\beta}$
1. Randomly initialize the coefficient vector $\boldsymbol{\beta}$
2. While $\boldsymbol{\beta}$ is not optimal
3. **for** $i = 1, 2, \ldots, N$ **do**
4. $\hat{\beta}_i \leftarrow \beta_i$
5. $\hat{G} = 2 \times \sum_{j=1}^{N} \beta_j K(\mathbf{x}_j, \mathbf{x}_i)$
6. $G \leftarrow \hat{G} + (\beta_i - \hat{\beta}_i) - 1$
7. $PG = \begin{cases} G & \text{if } 0 < \beta_i < C \\ \min(G, 0) & \text{if } \beta_i = 0 \\ \max(G, 0) & \text{if } \beta_i = C \end{cases}$
8. **if** $|PG| \neq 0$ **then**
9. $\beta_i \leftarrow \min(\max(\beta_i - \frac{1}{2}G, 0), C)$
10. **end if**
11. **end for**

3.2. Cluster labeling in novel ways

From the principle described in Section 2.2 and steps shown in Algorithm 2, to find out the connectivity status, we have to calcu-late $f(\mathbf{x}_{\tilde{m}})$ by using Eq. (3). To ease the burden, we propose a novel cluster labeling strategy to reach a stable and low average sample rate m. For efficiency, meanwhile, both of the number of sampled point-pairs N_m and the average sample rate m are significantly reduced.

Algorithm 2 Description of FRSVC.

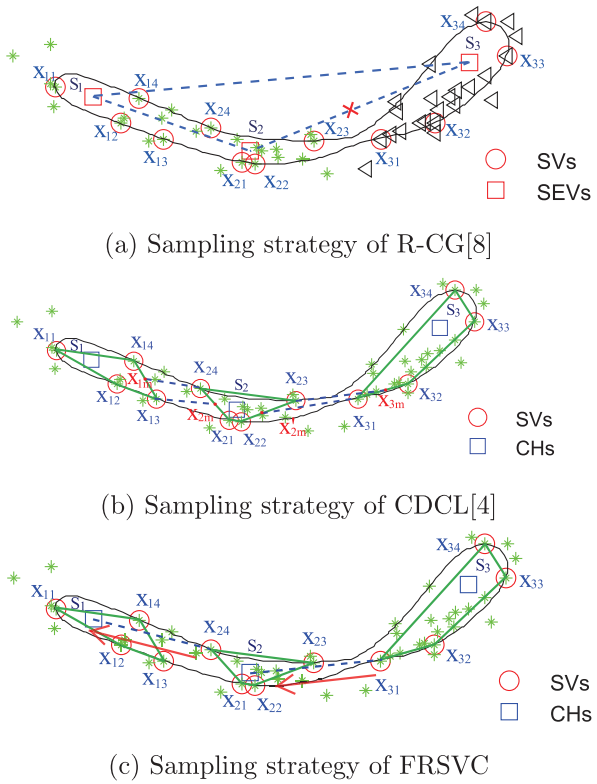
Require: Dataset $\mathcal{X}$, kernel width q, and penalty C
Ensure: Clustering labels for all the data samples
1. $\{\mathcal{X}_S, R, \boldsymbol{\beta}\} \leftarrow$ CollectSVsbyDCDSolver$(\mathcal{X}, q, C)$
2. $\boldsymbol{A} \leftarrow$ ConnAnalysisbyOnceSamp$(\mathcal{X}_S, R)$
3. Labels $\leftarrow$ FindConnComponents$(\boldsymbol{A})$
4. **for each** $\mathbf{x} \in \mathcal{X} \backslash \mathcal{X}_S$ **do**
5. $inx \leftarrow$ find the nearest SV from $\mathbf{x}$
6. Labels[$\mathbf{x}$] $\leftarrow$ Labels[$\mathbf{x}_{inx}$]
7. **end for**
8. **return** Labels

3.2.1. Group SVs by index of convex hulls

In practical, using SVs to find out the number of clusters and to label them firstly is considered as an effective way. Not only because the size is generally much smaller than the full size, but also for their excellent ability of profiling clusters. Our previous work [4] had proved that, SVs are essential and sufficient for profiling clusters which might be decomposed into a number of convex hulls, and proposed a convex decomposition based cluster labeling method (CDCL). By introducing this idea, in this study, we group SVs with respect to different convex hulls. For each convex hull, the surrounded SVs can converge to the same SEV based on a gradient dynamical system $\partial \mathbf{x}/\partial t = -\nabla f(\mathbf{x})$ [8] which has the local minimum distance, in feature space, to the hypersphere center $\boldsymbol{\alpha}$. Based on SEVs, we actually obtain a number of convex hulls which can be considered as cluster prototypes or exemplars. Even though one cluster might consist of one or multiple convex hulls, those SVs comprise of each convex hull can be grouped with the same label.

3.2.2. Connectivity analysis between two nearest neighboring convex hulls

As discussed in Ref. [4], convex hulls cannot enclose all of the data samples without pre-setting $C = 1$. Thus, connectivity analysis among convex hulls is to check whether any two convex hulls



(a) Sampling strategy of R-CG[8]

(b) Sampling strategy of CDCL[4]

(c) Sampling strategy of FRSVC

Fig. 1. Connectivity analysis with different strategies on partial figure of `ring` [8] ($q = 2, C = 0.1$).

are extracted from one cluster. Intuitively, two nearest neighboring convex hulls should be firstly taken into account. Then, merging them possible in one cluster from bottom to top is adopted in this study.

Fig. 1 compares our connectivity analysis strategy with two classic strategies, i.e., reduced complete graph (R-CG) [8] and CDCL [4]. It shows a partial result of clustering `ring` [8] done by R-CG, CDCL and FRSVC separately. Although R-CG and CDCL use similar method to group SVs, distinguished way of connectivity analysis can be concluded respectively as follows:

(1) Taking SEV as the cluster prototype, R-CG samples segmers on line segment connecting all the SEV pairs. Suppose that there are N_c SEVs, then, the number of sampled line segment N_m is $\frac{1}{2}N_c(N_c - 1)$. However, SEVs are inner samples. From one SEV to another SEV, the line segment can hardly reflect the connectivity relationship of two neighboring clusters, it might cross outside the cluster boundary and return an incorrect result (Fig. 1a).

(2) Taking convex hull as the cluster prototype, CDCL samples segmers on line segment which connects two nearest neighboring convex hulls and cross the utmost overlap region. For each line segment, thus, one side is a SV, the other side is the intermediate point of the nearest two SVs (from the neighboring convex hull). For instance, we can find line segments $\overline{\mathbf{x}_{13}\mathbf{x}_{2m}}$, $\overline{\mathbf{x}_{1m}\mathbf{x}_{24}}$, $\overline{\mathbf{x}_{22}\mathbf{x}_{3m}}$, $\overline{\mathbf{x}_{31}\mathbf{x}'_{2m}}$. Although it might require more line segments, $N_m = N_c(N_c - 1)$ for the worst case, with a disconnection checking first strategy to reduce m, CDCL still reaches an comparable efficiency while achieves high accuracy (Fig. 1b).

Different from R-CG and CDCL, the basic idea of connectivity analysis behind FRSVC is quite simple: Since SVs locate on the cluster boundary, any line segment starting from a SV, if it don't cross the convex hull which it belongs to, to its nearest neighbor-

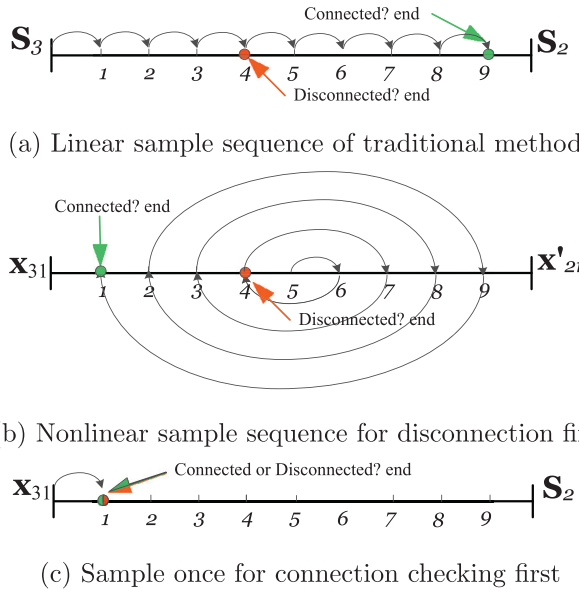


Fig. 2. Comparison of sample sequence generated by different strategies.

ing convex hull, will have an extremely high possibility of crossing outside the expected cluster at the first beginning. Therefore, to analyze the connectivity of two nearest neighboring convex hulls, is to check whether there is a path for a SV moving to the lowest place (i.e., SEV) of its nearest neighboring convex hull.

Notice that what we only need to do is to verify if the expected path exists. Thus, as depicted by Fig. 1c, there are three convex hulls represented by S_1, S_2 and S_3, respectively. Obviously, S_1, S_2 and S_2, S_3 are two pairs of nearest neighboring convex hulls. Taking S_2 and S_3 for example, $\mathbf{x}_{31}$ is a SV of S_3, and which is the nearest to S_2. So, checking the line segment starting from $\mathbf{x}_{31}$ to S_2 is suitable for judging the connection status of S_2 and S_3. If and only if we unfortunately meet a disconnection status, we turn around to check line segment starting from $\mathbf{x}_{23}$ to S_3. Similarly, line segment $\overline{\mathbf{x}_{24}S_1}$ starting from $\mathbf{x}_{24}$ will be checked for extracting the connection status of S_2 and S_1. By employing the proposed method, the number of line segments N_m required to be checked ranges from $N_c - 1$ to $2N_c$ if there are N_c convex hulls.

3.2.3. Sample once for connection checking first

Consider the connectivity analysis between S_2 and S_3 in Fig. 1a, to check the connectivity of two clusters, the conventional methods [2,5,8,9,13–18] adopted a linear sample sequence, i.e., points sampled uniformly from one end to the other end on a line segment. Corresponding to Fig. 2a, it means that if the line segment $\overline{S_3S_2}$ is generated with 10 equipartitions (usually be in $10\sim 20$), they will check segmers from 1 to 9 orderly. Once a segmer $\mathbf{x}_{\tilde{m}}$ with $f(\mathbf{x}_{\tilde{m}}) > R^2$ is found, e.g., $\tilde{m} = 4$ in Fig. 2a, clusters represented by S_3 and S_2 are judged to be disconnected; otherwise, they make a connected conclusion only until the last segmer ($\tilde{m} = 9$) has been checked with $f(\mathbf{x}_{\tilde{m}=9}) \le R^2$. Undoubtedly, it frequently requires a high average sample rate. By introducing convex decomposition method, Ref. [3,4] found that the farther from associate external location to convex hulls is, the greater the probability of the corresponding local region to be sparse distribution with data point is. So they proposed a nonlinear sample sequence for disconnection first. Using the same case with Fig. 2a, the center (i.e., $\tilde{m} = 5$) of the line segment $\overline{\mathbf{x}_{31}\mathbf{x}'_{2m}}$ is selected as the initial point and be checked first (See Fig. 2b). If two convex hulls are strongly disconnected, or even weak disconnected (detailed by Ref. [4]), the checking procedure will be ended quickly. Less than 3 sgemers are usually enough for getting the final decision correctly. However, in the worst case, i.e., two convex hulls are strongly connected, the full sample sequence of $\tilde{m} = \{6, 4, 7, 3, 8, 2, 9, 1\}$ will be checked before it ends. For accuracy, if $\mathbf{x}_{31}$ and $\mathbf{x}'_{2m}$ are judged being disconnected in the first round, we turn around to do an additional analysis on the other line segment $\overline{\mathbf{x}_{23}\mathbf{x}_{3m}}$. As analyzed by Ref. [1], nonlinear sample sequence for disconnection first requires much smaller m than that needed by the traditional methods.

Based on the basic idea in Section 3.2.2, the proposed sampling strategy is to see whether there is a path for a SV moving to the lowest place of its nearest neighboring convex hull. As described by Fig. 2c, connectivity analysis is done between two nearest neighboring convex hulls, and the sampling direction is from SV $\mathbf{x}_{31}$ of convex hull S_3 to SEV S_2. No matter the distance from the first segmer to the hypersphere center $f(\mathbf{x}_{\tilde{m}=1})$ is greater or lower than R^2, sampling in this direction will be ended. If and only if $f(\mathbf{x}_{\tilde{m}=1}) > R^2$, an additional check for $f(\mathbf{x}_{\tilde{m}=1})$ on line segment connecting $\overline{\mathbf{x}_{23}S_3}$ will be done. That means sampling twice is only for the worst case, and the connected conclusion can avoid an additional once check from the opposite direction. Therefore, an significant reduction of the average sample rate is achieved.

3.3. The basic framework of FRSVC

Compared with problem (2), the constraint condition of (8) is slacked such that we have more choices of methods to solve it. Although it can be solved efficiently, however, we can hardly guarantee the radius $R < 1$. So, for stable performance, we present a basic framework of FRSVC in Algorithm 2 following a similar form with the classic SVC.

Given q and C, `CollectSVsbyDCDSolver`$(\cdot)$ in line 1 obtains $\boldsymbol{\beta}$ by solving (8) in terms of DCD solver, collects SVs $\mathcal{X}_S$ with $0 < \beta_i < C (i \in [1, N])$, and estimates the sphere radius R according to square root of Eq. (3). Then, in line 3, `ConnAnalysisbyOnceSamp`$(\cdot)$ employs the sampling strategy presented in Section 3.2 to check the connectivity of two nearest neighboring convex hulls which are constructed by groups of SVs. Based on a simple but effective idea, it only checks the first segmer on line segment starting from SV to convex hull. For efficiency, only when meets a disconnected judgment, it turns around for an additional once check. After that, the adjacency matrix $\boldsymbol{A}$ is obtained for immediate clusters discovering by `FindConnComponents`$(\cdot)$. The output is an array with size N_S which contains the cluster labels. Finally, the remaining data samples are separately assigned with the labels of their nearest SVs.

4. Performance analysis

4.1. Time complexity

To measure the time complexity of FRSVC, let N be the number of samples in a data set, N_{SV} be the number of SVs, ℓ be the average number of iterations for each data sample to locate its corresponding local minimum via the steepest decent process [8], N_c be the final number of convex hulls, and m be the average sample rate.

For the training phase, as described in Algorithm 2, the pre-computed kernel matrix is not prerequisite. Whether to use it for efficiency at the cost of storage or to calculate corresponding row of kernel values on demand (line 5 of Algorithm 1), depends on the actual memory capacity. If with sufficient memory, i.e., space complexity of $O(N^2)$ to store the kernel matrix, the computational complexity required by the data owner in training phase is $O(N^2)$; otherwise, it can be up to $O(dN^2)$. Here, d is the data dimension. Since the pre-computed kernel matrix can partially employed, in practical approach, the actual computational complexity of the train-

Table 1
Comparisons of time complexity.

Index	Method	SVC training	Labeling
1	CG	$O(N^3)$	$O(mN^2)$
2	R-CG	$O(N^3)$	$O(\ell N + mN_c^2)$
3	E-SVC	$O(N^3)$	$O(\ell N + mN_c^2 + 2\ell N_c)$
4	CCL	$O(N^3)$	$O(N_{SV}^2)$
5	FSVC	$O(N^3)$	$O(\ell N_b + \gamma N^2)$
6	PSVC	$O(N^3)$	$O(mN^2)$
7	CDCL	$O(N^3)$	$O(\ell N_{SV} + 2mN_c)$
8	VCC	$O(N_{tr}^3)$	Mode I: $O(\ell N_{tr} + mN_c^2)$ Mode II: $O(\ell N_b + \gamma N_{tr}^2 + mN_b^2)$
9	FSSVC	$O(N^2 + M^3)$	$O(\ell N_{SV} + 2mN_c)$ or $O(N_{SV}^2)$
10	FRSVC	$O(\tilde{d}N^2)$	$O(\ell N_{SV} + \tilde{m}N_c)$

Note: $\tilde{d} \in [1 \le \tilde{d} \le d]$, $\tilde{m}(\tilde{m} \in [1, 2])$; $N_{tr} = \theta N$, θ is the sample rate in $(0, 1]$.

Table 2
Description of the benchmark data sets.

Data sets	Data set description		
	Dims	Size	# of classes
`Orange`	2	140	9
`Sunflowers`	2	200	9
`Twocircles`	2	300	2
`Iris`	4	150	3
`Wisconsin`	9	683	2
`Wine`	13	178	3
`Zoo`	16	101	7
`movement_libras (mLibras)`	90	360	15
`five-Gaussians (f-Gaus)`	2	1000	5
`D31`	2	3100	31
`P2P traffic (P2P-T)`	4	9206	4
`WebKB`	4	4199	4
`Reuters-21578(Reuters)`	10	9990	10
`Ohsumed (Oh)`	23	13,929	23
`20Newsgroups (20NG)`	20	13,998	20
`Shuttle (sh)`	9	43,500	7

ing phase is $O(\tilde{d}N^2)$ where $\tilde{d} \in [1 \le \tilde{d} \le d]$. Apparently, it is much lower than $O(N^3)$ required by the conventional methods[1]. In the labeling phase, the time cost for constructing convex hulls and connectivity analysis depend on N_{SV} and the actual number of convex hulls N_c which is normally much lower than N_{SV}. By employing the proposed strategy, the exact times of sampling between two nearest neighboring convex hulls is uncertain, which can be 1 or 2. For the simplicity, we use $\tilde{m}(\tilde{m} \in [1, 2])$ to represent it. Cluster assignment of the convex hulls and cluster matching for each takes $O(N_c)$ and $O(N)$ of elementary calculation, respectively, which are ignorable in practice. Thus, FRSVC consumes $O(\ell N_{SV} + \tilde{m}N_c)$ to finish the labeling phase.

Furthermore, in Table 1, we compare FRSVC with the baseline strategy of complete graph (CG) [5] and its state-of-the-art variants, i.e., R-CG [8], equilibrium based SVC (E-SVC) [9,17,19,20], cone cluster labeling (CCL) [21,22], fast support vector clustering (FSVC) [2], position regularized support vector clustering (PSVC) [23], CDCL [4], voronoi cell-based clustering (VCC) [24], and fast and scalable SVC (FSSVC) [3]. In Table 1, M is the size of the selected cluster boundaries (see Ref[3] for the details), γ ranges from $1/N$ to 1, N_{tr} is the number of data which is uniformly sampled with a predefined sample rate θ, and N_b is the number of small balls extracted from either N_{tr} data samples for VCC or the whole data set for the others. Even though two optional modes of labeling phase are presented for VCC, in this study, Mode I with fast phase to label the remaining data is preferred for its relative stability. From the comparisons, a flexible time complexity of FRSVC can be found that reflects its balance strategy between efficiency and availability for diverse platform.

4.2. Datasets and experimental settings

Since the proposed DCD solver do not change the fundamental principle of model training, in the experiments, we first compare the computational time of training phase between CG [5] and FRSVC. Taking CG as the baseline is because its training way has been inherited by most of the state-of-the-art variants. In practical, the size of SVs and the average sample rate are two major factors related to the efficiency. So we conduct experiments on sixteen data sets to collect time consumptions in training and labeling corresponding to the former. The latter is measured by comparing with three representative strategies in Section 4.5. To evaluate the accuracy, adjusted rand index (ARI, expressed by Eq. (19)) [25] which is a widely used similarity measure between two data partitions where both true labels and predicted cluster labels are given. In Eq. (19), N_{ij} is the number of data points with true label i but they are assigned by j, $N_{i\cdot}$ and $N_{\cdot j}$ are the number of data points with label i and j respectively. Due to the existing criteria with respect to accuracy for clustering do not consider the cluster number, which is an important factor to avoid achieving great accuracy by splitting a data set into amount of clusters, the obtained cluster number is also taken into account in this study. Finally, we compare the average sample rate achieved by the proposed method and the state-of-the-art methods who adapt either linear or nonlinear sampling sequence for efficiency. It will verify if the proposed strategy of sample once for connection checking first contributes to the efficiency.

$$\mathrm{ARI} = \frac{\sum_{i,j}\binom{N_{ij}}{2} - \left[\sum_i \binom{n_i}{2}\sum_j \binom{N_{\cdot j}}{2}\right]/\binom{N}{2}}{\frac{1}{2}\left[\sum_i \binom{N_i}{2} + \sum_j \binom{N_{\cdot j}}{2}\right] - \left[\sum_i \binom{N_i}{2}\sum_j \binom{N_{\cdot j}}{2}\right]/\binom{N}{2}} \tag{19}$$

The aforementioned experiments are conducted on various data sets: `sunflowers`, `five-Gaussians`, `orange`, `twocircles` and `D31` are used in [2,10,26,27] and `iris`, `wisconsin`, `wine`, `zoo`, `movement_libras` and `shuttle` are from UCI repository [28]. In addition, four text corpora are employed after a preprocessing namely $\mathrm{DC}_{\mathrm{GLI}}$-CCE by [29], i.e., four categories of `WebKB` [30], full twenty categories of `20Newsgroups` [31], top 10 largest categories of `Reuters-21578` [32] and `Ohsumed` with 23 classes [33]. `P2P traffic` is a collection of 9206 flows' features which are extracted from traffic supplied by [34] following the method of [35].

The statistics of sixteen data sets are listed in Table 2. Due to space limitation, we use abbreviations in brackets for data with long name. The program of FRSVC are written in MATLAB. As the previous settings, usability of the proposed method is the crucial concern in this study. We do not try to find out the computational time savings brought by powerful CPU, because this aspect of improvement on efficiency is a matter of course. Therefore, the testbed features with computers, one with Intel Dual Core 2.66 GHz and 32GB RAM (named Server for supplying sufficient storage for the pre-computed kernel matrix), and the other with the same CPU clock speed yet with 3GB RAM (named Client which can not afford the pre-computed kernel matrix). Both of them run Windows 7-X64 and MATLAB 2014a.

4.3. Performance contrast with the classic SVC

Notice that memory size is the major difference between the experimental platforms. Generally, they will not make significant difference on computational time if they both meet the storage requirement of kernel matrix. However, Client with relatively limited memory size might not able to afford the storage loads for some cases, e.g., `Oh` and `sh`. Fortunately, as the previous discuss, the proposed method allows immediate calculation for rows of kernel ma-

Table 3
Time comparison of the training phase with/without the pre-computed kernel matrix (s.).

Data sets	CG	FRSVC	
	(Client)	(Client)	(Server)
Orange	**0.02**	0.12	0.03
Sunflowers	**0.02**	0.07	0.03
Twocircles	0.05	0.09	**0.03**
Iris	**0.01**	0.77	0.06
Wisconsin	0.05	6.45	**0.03**
Wine	**0.02**	0.37	0.04
Zoo	**0.01**	0.76	**0.01**
mLibras	**0.05**	60.02	0.15
f-Gaus	2.62	0.91	**0.11**
D31	45.19	8.88	**0.66**
P2P-T	278.37	67.74	**0.29**
WebKB	4.87	7.07	**0.41**
Reuters	58.31	13.47	**2.56**
Oh	85.09*	52.71	**13.73**
20NG	12.62*	45.45	**10.14**
sh	867.16*	203.48	**1.68**

trix and distributed storage of kernel matrix by row. For the consistency, therefore, the evaluated FRSVC on Client employs immediate calculation, while FRSVC executed on Server will store the pre-computed kernel matrix for value retrieval and computation for speeding up the DCD solver.

Table 3 illustrates the time comparison of training phase with/without using the pre-computed kernel matrix. Time consumption of each method is an average value of ten times of the execution on each data set. In particular, the first rank with the smallest time-consumption for each data is highlighted by boldface. Notice that Oh, 20NG and sh are too large for Client to construct the full kernel matrix, thus the evaluation of CG on these three data sets was conducted on Server. The corresponding results are marked by '*' as superscript for distinguishing. According to the performance described in Table 3, we can find some interesting properties as follows:

(1) An obvious separatrix can be found between mLibras and f-Gaus. For data sets in rows before mLibras, CG requires the least time for most cases, i.e., orange, sunflowers, iris, wine, zoo and mLibras. The related data sets are small enough for Client to do pre-computation and store the full kernel matrix. However, for all the other data sets, FRSVC on Server outperforms both CG and FRSVC on Client, because the efficiency of the DCD solver and it uses items of the kernel matrix in direct, respectively. It confirms that the proposed DCD solver flexibly handles data in any scale well.
(2) Although the training phase of CG was done by Server for Oh, 20NG and sh, it is much slower than on Server. On the basis of the time complexity analysis in Section 4.1, it is because the proposed DCD solver uses simpler computations and needs less iterations ℓ to obtain a right coefficient vector for large-scale data set than that of required by CG. In addition, as the data size increases, two interesting observations can be found. Firstly, the time-consumptions of CG and FRSVC on Server are similar when they cluster small data sets, since not too many iterations are required by their solvers. Secondly, the actual time cost is also effected by data distribution which is directly related to the separability (closely related to accuracy discussed in the next section). For instance, the size of sh is much greater than that of 20NG, however, FRSVC on Server consumes much less on sh. Similar cases can be found on P2P-T and Oh by CG.
(3) Due to calculating the required kernel values in each iteration, FRSVC on Client consumes much more time than FRSVC on Server, especially on data sets with high dimensions (e.g., mLibras). But even so, it outperforms CG for many cases, i.e., f-Gaus, D31, P2P-T, Reuters, Oh and sh. Undoubtedly, the reduced iterations benefit it and compensates for the lost of efficiency caused by calculation. More importantly, by trading time for space, FRSVC on Client works well on large-scale data even with limited memory.

4.4. Performance contrast with the state-of-the-art methods

To make a deep analysis, we conduct experiments on sixteen data sets described by Table 2. The state-of-the-art SVC methods namely CG [5], R-CG [8], E-SVC [9,17,19,20], CCL [21,22], FSVC [2], PSVC [23], CDCL [4], VCC [24] and FSSVC [3] are introduced. There performances on these data sets are collected and compared with the proposed FRSVC.

Towards achieving full comparisons, we include time costs for training phase and labeling phase separately, the number of obtained clusters, and ARI achieved by each method. All the experimental results are listed in Table 4. Rank of each one is given depending on its performance measure followed by the corresponding rank (from 1 to 3). In particular, the value of rank 1 for each test item is highlighted by boldface. Three points are important to be noted. First, the time cost is an average value of ten times of the execution on each data. And no clear rank marked for training phase except for boldface for the first rank, because the major reason for difference in training phase, for methods excluding CCL, FSSVC and FRSVC, is the parameter settings. Secondly, due to the challenge of sampling strategy, VCC can not fixedly achieve accuracy even though its parameters are fixed. Thence, based on far more than ten results by fixing the optimal parameters, its mean and mean-square deviation of the top ten ARIs are calculated for presentation. For fairness, the mean is used for ranking. Thirdly, for large-scale data sets, e.g., sh, not all the evaluated methods can finish analysis on Client while the kernel matrix is too large for Client to afford. Hence, for fairness, the training phases of all the methods are carried out on Server, and the labeling phases are kept on Client. However, we still encounter that some methods require more than 3 hours to complete either training or labeling. For these cases, we omit them or use "—" to denote an unavailable result.

Based on data characteristics in Table 2, from results shown in Table 4, some conclusions are revealed as follows:

(1) In terms of ARI measure, it is apparent that FRSVC achieves the best results on most of data sets (namely sunflowers, twocircles, zoo, f-Gaus, D31, WebKB, 20NG, Oh and sh), especially for the large-scale ones, whereas FSSVC performs better for orange, wisconsin, wine and mLibras, and CDCL gets better performances on iris, P2P-T and Reuters. Furthermore, for some small data, e.g., zoo, CG, R-CG and E-SVC obtain comparable results with FRSVC, FSSVC or CDCL, while VCC does that on some of large-scale data, e.g., 20NG and Oh. Unfortunately, R-CG can hardly clusters Reuters well since seriously overlap regions exist due to the special vectorizing way of [29]. Although FRSVC can not always reach the best performance, apparently, it is consistently included in the first three ranks, and the achieved results are comparable with the corresponding first two and the differences are not significant. They, at least, support that FRSVC is effective, especially its labeling strategy can accurately capture the distribution characteristics of dataset. Based on ARI measures, we also give results of pair comparisons in Table 5 following the work of [36]. Here, FRSVC is the control method. A nonparametric statistical test of Friedman test [37] is employed to get the average ranks and

Table 4
Benchmark results on various datasets.

Data	Methods	Training (Server)	Labeling (Client)	$N_R/(N_C)$	*ARI*
Orange	CG	0.02	1.29	21/(9)	0.7478
	R-CG	**0.01**	1.06	**9**/(9)	*0.9282*[c]
	E-SVC	**0.01**	15.25	**9**/(9)	0.9090
	CCL	0.57	*0.21*[c]	5/(9)	0.4993
	FSVC	**0.01**	0.78	**9**/(9)	0.8848
	PSVC	0.06	1.22	22/(9)	0.7478
	CDCL	**0.01**	**0.17**[a]	**9**/(9)	*0.9283*[b]
	VCC	0.02	9.05	**9**/(9)	0.8094 ± 0.0521
	FSSVC	0.04	*0.18*[b]	**9**/(9)	**0.9312**[a]
	FRSVC	0.03	0.46	**9**/(9)	*0.9283*[b]
Sunflowers	CG	**0.02**	3.09	**9**/(9)	**1.0000**[a]
	R-CG	**0.02**	1.49	**9**/(9)	**1.0000**[a]
	E-SVC	**0.02**	106.24	10/(9)	*0.9866*[b]
	CCL	1.13	**0.17**[a]	10/(9)	0.8676
	FSVC	0.03	0.67	**9**/(9)	*0.9807*[c]
	PSVC	0.10	3.55	**9**/(9)	**1.0000**[a]
	CDCL	**0.02**	*0.76*[c]	**9**/(9)	**1.0000**[a]
	VCC	**0.02**	6.11	**9**/(9)	0.8264 ± 0.1026
	FSSVC	0.07	0.78	**9**/(9)	**1.0000**[a]
	FRSVC	0.03	*0.28*[b]	**9**/(9)	**1.0000**[a]
Twocircles	CG	0.05	9.96	**2**/(2)	**1.0000**[a]
	R-CG	0.06	2.87	4/(2)	0.6770
	E-SVC	0.21	28.40	4/(2)	*0.7355*[c]
	CCL	8.67	1.04	**2**/(2)	*0.7619*[b]
	FSVC	**0.02**	*0.73*[c]	19/(2)	0.1402
	PSVC	0.61	14.16	**2**/(2)	**1.0000**[a]
	CDCL	0.10	*0.67*[b]	**2**/(2)	**1.0000**[a]
	VCC	0.03	9.70	4/(2)	0.9480 ± 0.0361
	FSSVC	0.06	**0.37**[a]	**2**/(2)	**1.0000**[a]
	FRSVC	0.03	2.09	**2**/(2)	**1.0000**[a]
Iris	CG	**0.01**	1.12	11/(3)	0.6178
	R-CG	**0.01**	3.06	16/(3)	0.7374
	E-SVC	**0.01**	4.73	2/(3)	0.5681
	CCL	0.04	*0.72*[c]	**3**/(3)	*0.8858*[b]
	FSVC	**0.01**	1.06	5/(3)	0.5620
	PSVC	0.06	1.44	27/(3)	0.6147
	CDCL	**0.01**	*0.66*[b]	**3**/(3)	**0.9222**[a]
	VCC	0.02	2.29	7/(3)	0.8141 ± 0.0129
	FSSVC	0.03	**0.12**[a]	6/(3)	*0.8509*[c]
	FRSVC	0.06	*0.72*[c]	**3**/(3)	*0.8858*[b]
Wisconsin	CG	0.05	319.58	8/(2)	0.7793
	R-CG	**0.03**	22.21	13/(2)	0.8035
	E-SVC	0.04	443.20	46/(2)	0.1344
	CCL	100.66	23.69	**2**/(2)	*0.9076*[b]
	FSVC	0.04	13.02	153/(2)	0.6687
	PSVC	2.56	190.38	109/(2)	0.2574
	CDCL	**0.03**	*1.28*[b]	**2**/(2)	*0.8685*[c]
	VCC	0.12	2.27	**2**/(2)	$0.8029 \pm$ -0.0514
	FSSVC	0.23	**0.89**[a]	**2**/(2)	**0.9248**[a]
	FRSVC	**0.03**	*2.24*[c]	**2**/(2)	*0.8798*[c]
Wine	CG	0.02	2.98	31/(3)	0.5912
	R-CG	**0.01**	7.78	20/(3)	0.7928
	E-SVC	**0.01**	36.70	12/(3)	0.4159
	CCL	0.80	**0.63**[a]	**3**/(3)	0.8190
	FSVC	0.02	7.60	15/(3)	0.8042
	PSVC	0.09	3.46	28/(3)	0.3809
	CDCL	**0.01**	*0.80*[c]	**3**/(3)	*0.8961*[b]
	VCC	0.02	2.18	**3**/(3)	0.8088 ± 0.0563
	FSSVC	0.08	0.90	**3**/(3)	**0.8992**[a]
	FRSVC	0.04	*0.78*[b]	**3**/(3)	*0.8483*[c]
Zoo	CG	**0.01**	0.60	**8**/(7)	*0.9342*[c]
	R-CG	**0.01**	3.81	**8**/(7)	**0.9570**[a]
	E-SVC	**0.01**	19.84	**8**/(7)	**0.9570**[a]
	CCL	0.52	*0.52*[b]	12/(7)	0.8343
	FSVC	**0.01**	2.66	10/(7)	0.8679
	PSVC	0.04	*0.69*[c]	9/(7)	0.7441
	CDCL	**0.01**	2.82	**6**/(7)	*0.9469*[b]
	VCC	0.02	12.87	10/(7)	0.9184 ± 0.0134
	FSSVC	0.03	**0.27**[a]	9/(7)	*0.9342*[c]
	FRSVC	**0.01**	3.80	**8**/(7)	**0.9570**[a]
mLibras	CG	0.05	15.08	139/(15)	0.2422
	R-CG	0.05*	252.84	138/(15)	0.2356
	E-SVC	—	—	—	—
	CCL	25.88	**1.03**[a]	230/(15)	0.0899
	FSVC	**0.02***	226.07	213/(15)	0.1421
	PSVC	0.29	11.89	104/(15)	0.2541
	CDCL	**0.04**	78.53	37/(15)	*0.3320*[c]
	VCC	0.13	443.03	36/(15)	0.3065 ± 0.0181
	FSSVC	0.53	*3.67*[b]	48/(15)	**0.3703**[a]
	FRSVC	0.15	*10.75*[c]	**32**/(15)	*0.3366*[b]
f-Gaus	CG	2.62	86.39	17/(5)	0.4712
	R-CG	3.09	10.89	17/(5)	*0.8693*[c]
	E-SVC	3.09	1079.60	18/(5)	0.8585
	CCL	300.48	413.66	19/(5)	0.0021
	FSVC	1.34	*1.04*[b]	17/(5)	0.7137
	PSVC	4.37	745.83	42/(5)	0.0012
	CDCL	3.09	*4.82*[c]	11/(5)	*0.8807*[b]
	VCC	**0.11**	11.44	16/(5)	$0.7550 \pm$ 0.1525
	FSSVC	0.26	**0.93**[a]	7/(5)	*0.8704*[c]
	FRSVC	**0.11**	7.80	**5**/(5)	**0.9457**[a]
D31	CG	45.19	3548.69	10/(31)	0.2345
	R-CG	5.51	47.76	41/(31)	0.8653
	E-SVC	—	—	—	—
	CCL	—	—	—	—
	FSVC	0.68	**1.28**[a]	**31**/(31)	0.5611
	PSVC	78.08	6962.92	195/(31)	0.4518
	CDCL	4.39	*15.39*[c]	45/(31)	*0.9020*[b]
	VCC	**0.14**	34.34	**31**/(31)	0.8252 ± 0.0421
	FSSVC	0.91	*1.57*[b]	32/(31)	*0.8767*[c]
	FRSVC	0.66	29.62	**31**/(31)	**0.9429**[a]
P2P-T	R-CG	276.96	199.33	5/(4)	*0.8815*[b]
	FSVC	1066.17	*8.01*[b]	**4**/(4)	*0.8367*[b]
	CDCL	268.08	11.27	**4**/(4)	**0.8917**[a]
	VCC	0.39	25.15	5/(4)	0.7389 ± 0.0066
	FSSVC	14.87	**1.02**[a]	4/(4)	*0.8815*[b]
	FRSVC	**0.29**	*10.11*[c]	**4**/(4)	*0.8678*[c]
WebKB	R-CG	2.64	177.75	171/(4)	0.3072
	FSVC	13.84	20.55	28/(4)	*0.5144*[c]
	CDCL	1.04	*5.81*[b]	7/(4)	0.4645
	VCC	**0.16**	*9.89*[c]	**4**/(4)	0.4434 ± 0.0156
	FSSVC	1.42	**3.93**[a]	14/(4)	*0.5670*[b]
	FRSVC	0.41	12.31	**4**/(4)	**0.6395**[a]
20NG	VCC	**1.88**	*56.5661*[b]	27/(20)	0.4379 ± 0.0461
	FSSVC	81.58	**50.60**[a]	105/(20)	*0.3628*[b]
	FRSVC	10.14	*112.66*[c]	**26**/(20)	**0.4927**[a]
Oh	VCC	**4.63**	*92.76*[b]	**32**/(23)	0.4280 ± 0.0292
	FSSVC	73.01	**7.13**[a]	103/(23)	*0.4514*[b]
	FRSVC	13.73	*630.01*[c]	98/(23)	**0.4840**[a]
Reuters	R-CG	6.34	8315.19	2/(10)	0.0148
	FSVC	43.03	1351.14	1586/(10)	0.4775
	CDCL	31.39	506.86	88/(10)	**0.8064**[a]
	VCC	3.16	*13.43*[b]	**12**/(10)	0.4908 ± 0.0616
	FSSVC	12.35	**12.01**[a]	64/(10)	*0.5831*[c]
	FRSVC	**2.56**	*341.20*[c]	**12**/(10)	*0.7295*[b]
sh	E-SVC	—	—	—	*0.59*[c][2]
	FSVC	—	—	—	0.58[2]
	VCC	3.31	**29.33**[a]	14/(7)	0.5898 ± 0.0198
	FSSVC	608.90	*279.59*[c]	33/(7)	*0.6857*[b]
	FRSVC	**1.68**	*125.74*[b]	**13**/(7)	**0.8050**[a]

Note: [a]Rank 1, [b]Rank 2, [c]Rank 3.
"—" means not available or more than 3h. consumed.

unadjusted p values. By introducing an adjustment method Bergmann–Hommel procedure [38], the adjusted p-value denoted by p_{Homm} corresponding to each comparison is obtained. Obviously, FRSVC reaches the best performance in the view of average rank. Since the Bergmann-Hommel procedure rejects those hypotheses with p-values ≤ 0.025, together with the values of p_{Homm}, they further confirm that

Table 5
Comparison results under non-parametric statistical test.

Methods	Average ranks	Unadjusted p	p_{Homm}
control method : FRSVC, average rank = **2.0938**			
CG	7.0000	4.5745E-6	2.7447E-5
R-CG	5.5000	0.0015	0.0053
E-SVC	7.3125	1.0862E-6	7.6031E-6
CCL	7.4063	6.9437E-7	4.8606E-6
FSVC	6.8125	1.0421E-5	5.2105E-5
PSVC	7.5938	2.7754E-7	2.4979E-6
CDCL	3.4375	0.2094	0.4187
VCC	5.3125	0.0026	0.0079
FSSVC	2.5313	0.6828	0.6828

FRSVC performs better than FSSVC which significantly outperforms the others.

(2) For time consumption, the training phase and labeling phase are presented, separately, because they are considered as two potential computational bottlenecks of SVC. In the view of methodology, out of the ten methods, only VCC and FSSVC use a subset to work out the support function, while the others adopt the full data set. The difference is that FSSVC performs stably, because it selects a fixed subset of boundary data when the parameters are set. However, the boundary data can not be utilized by VCC. Although the time cost for data sampling and boundary data selection are introduced into the training phase of VCC and FSSVC, respectively, the corresponding computations in input space (not in feature space) might benefit efficiency. We consider it a fair comparison because the training phase and the labeling phase are separately measured, and Server has sufficient memory for the full kernel matrix to meet the requirement. In addition, we restrict the sample rate θ of VCC in the range of [0.001, 0.5) for data size greater than D31's to reach balance between accuracy and efficiency. For the training phase, some apparent observations can be found. 1) CG, R-CG, E-SVC and CDCL are grouped together for the same methodology used in solving the dual problem. Obviously, they cost similarly to complete the training phase. Nuances are due to their distinct parameter settings to cater themselves' labeling strategy towards the best ARI measure. Although CCL employs the same methodology, it requires more iterative analysis to guarantee $R < 1$. Therefore, along with the size increases, the efficiency of CCL reduces dramatically. 2) Exactly, FSVC is a fast method for most of cases. However, it occasionally consumes a lot on some categorical data, e.g., `P2P-T` and `WebKB`. 3) As a typical position weighted method, the pre-computed weights given by PSVC are meaningful. However, while these weights are additionally imposed constraints to the coefficients before solving the dual problem, the training time for PSVC is pricey. 4) When most of the evaluated methods fails, VCC, FSSVC and FRSVC are quite qualified for those large-scale data. As shown in column 3 of Table 4, in spit of much more data samples being taken into account by FRSVC, it performs comparably with VCC, and both of them outperform FSSVC frequently. An obvious reason may be that, the operations in the proposed DCD solver is quit simple, and not an amount of essential iterations required by the solver when the data set is large. By considering the aforementioned, undoubtedly, even a tiny difference existing between methods employed for training would generate diverse requirements of time consumption.

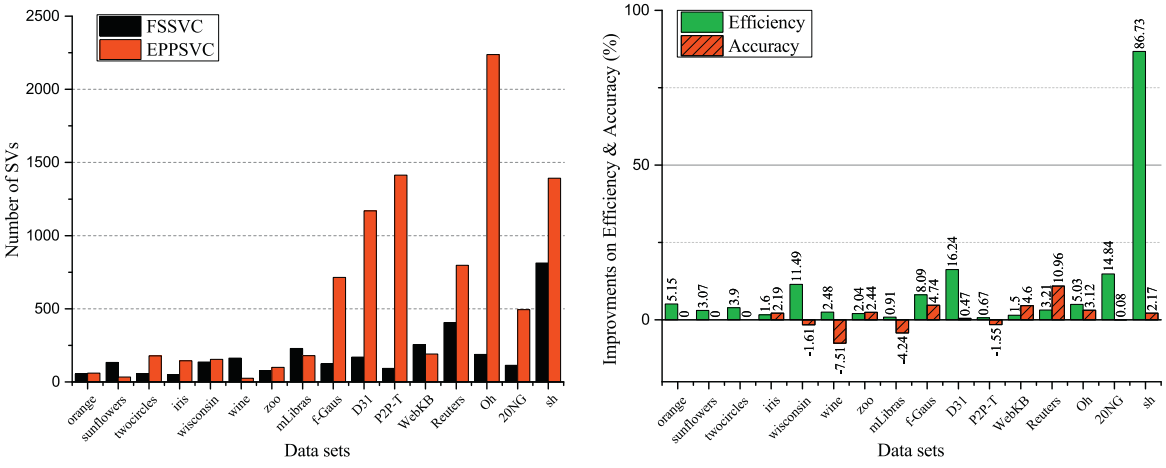
(3) Furthermore, for the labeling phase, FRSVC achieves the first three ranks for most cases. Although CG, R-CG, CCL and FSVC perform well in many cases, their labeling time increases dramatically as the dimensionality or size of data increases. Another time-consuming algorithm is E-SVC, which still fails in handling the 90-dimensional data set `mLibras`. Similar results can be found for PSVC. Therefore, they are more affordable for small and low dimensional data. Notice that VCC also performs well for most cases, especially on large-scale data, e.g., `sh`; whereas it suffers from high dimension, even though the data size is small, e.g, `mLibras`, `zoo` and `f-Gaus`. Among the ten evaluated methods, FSSVC is an exception. It performs stably and outperforms the others in most cases. From time complexity analysis in Table 1, the number of SVs may be the major contribution to efficiency. To confirm this, in Fig. 3a, we compare the numbers of SVs respectively obtained by FRSVC and FSSVC. Together with column 4 of Table 4, the proposed labeling strategy makes up the effect of larger SVs size left by the training phase of FRSVC than that of FSSVC, and it indeed achieve a comparable performance to FSSVC. Furthermore, we have tried to replace the labeling phase of FSSVC with the proposed labeling phase to further testify its performance, i.e., the standard FSSVC's training phase plus FRSVC's labeling phase. Improvements on both labeling efficiency and ARI measure for the sixteen data sets made by the proposed labeling phase over the standard FSSVC, in terms of percentage, are depicted by Fig. 3b. Clearly, efficiency improvements range from 0.67% to 86.73% and positive accuracy (ARI) improvements on 9 out of 13 data sets (the same ARIs obtained on the remaining 3 data sets) strongly confirm the effectiveness and usability of the proposed labeling strategy. Combining the description of data sets in Table 2, FRSVC is effective for multi-clusters data of different scales and has the superiority for clusters with different dimensionality.

(4) Intuitively, an effective method we expect should capture data distribution accurately. To estimate it, in this study, we use the discovered number of clusters which should be either equivalent to or close to the exact number of classes since the real clusters' number is unknown. In column 5 of Table 4, N_R denotes the discovered cluster number, while N_C is the exact number of classes summarized in Table 2. In clustering analysis, we should try to avoid splitting data samples of a group into different clusters [39]; thus, we highlighted the value of N_R by boldface if it is the closest to N_C. In this respect, the stability of FRSVC over different data sets can be seen from Table 4, where N_R obtained by FRSVC for 15 out of 16 data sets (except for `Oh`) outperforms all the other methods significantly.

4.5. Comparison of the sample rate

Every coin has two sides. Without a filtering mechanism to match DCD solver, FRSVC has to deal with much more SVs than that of obtained by FSSVC. Fortunately, the proposed sampling strategy (see Section 3.2.3) makes up the loss, such that FRSVC achieves a comparable labeling efficiency with FSSVC. Actually, no matter which solver is used to solving the dual problem, the ultimate objective is to select essential SVs from cluster boundaries [1,3], the employed labeling strategy based on the obtained SVs might be the first reason which responds to the efficiency and accuracy of labeling phase. So the interest of this experiment is to show how FRSVC reach our expected efficiency.

As introduced by Section 3.2.3, the sampling strategies of the evaluated methods in Section 4.4 can be grouped into three categories. The first category generates a linear sampling sequence between sampled point-pairs, e.g., CG, R-CG and PSVC, etc. In this section, we use "*linear in CG*" as the representative. The second category prefers either linear or nonlinear sampling sequences between two neighboring convex hulls to check their connectivity,



(a)Numbers of SVs obtained by FSSVC and FRSVC

(b)Improvements made by replacing the labeling phase of FSSVC with FRSVC's

Fig. 3. Comparison of the labeling phase between FSSVC and FRSVC.

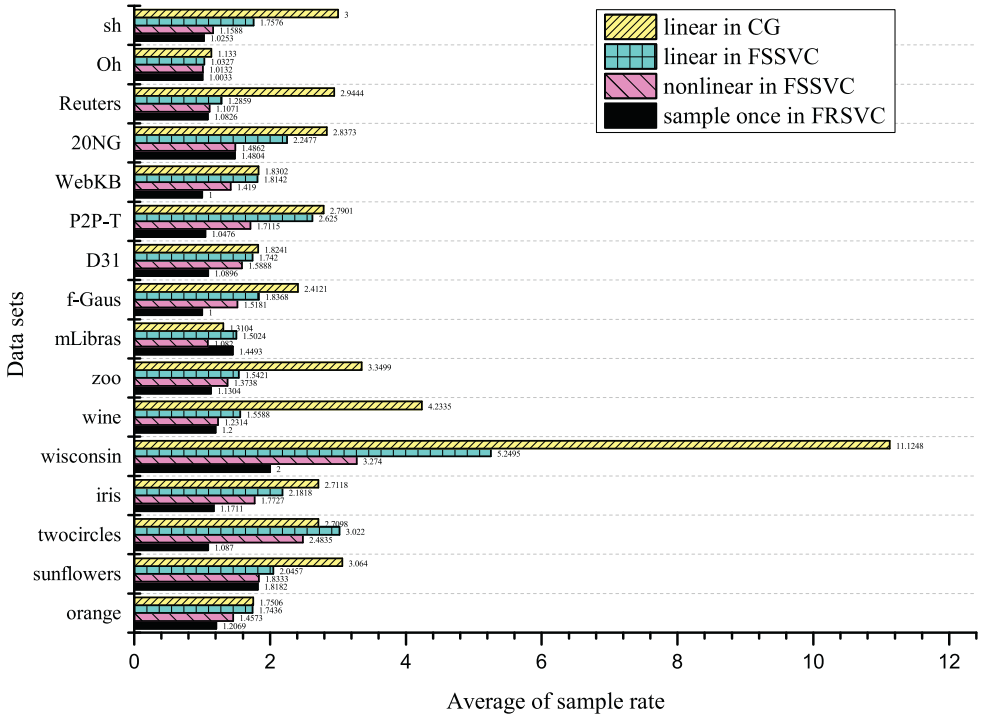


Fig. 4. Comparison of the average sample rates achieved by different strategies.

e.g., CDCL and FSSVC. To distinguish them, we take "*linear in FSSVC*" and "*nonlinear in FSSVC*" as their representatives, respectively. The proposed strategy is the third one, which is denoted by "*sample once in FRSVC*". On these sixteen data sets, the average sample rate required by the three types of strategies are illustrated by Fig. 4. By employing the proposed strategy , the average sample rate retains closely to one for absolutely most of data sets; whereas the other three requires much more. It means that there is frequently no need to turn around for an additional check between two nearest neighboring clusters. One exception is `wisconsin`, "*sample once in FRSVC*" reaches up to 2; whereas "*linear in CG*" checks 11.1248 segmers on the line segment, and both "*linear in FSSVC*" and "*nonlinear in FSSVC*" check more than 3 times. Totally, "*linear in FSSVC*" and "*nonlinear in FSSVC*" consistently perform better than "*linear in CG*". However, "*sample once in FRSVC*" outperforms them significantly except for `mLibras`. These comparisons precisely confirm the corresponding time-consumption listed in Table 4.

4.6. Further discussion on efficiency

As the aforementioned, in terms of both cluster validity and efficiency measures, FRSVC either outperforms the others significantly or achieves comparable result with the best one, especially on large scale data sets. Therefore, not only being of security, but also it is effective applicable for complicated data sets. However, efficiency potential is one of the most concerns. By considering analysis in Section 4.1 and evaluations in Table 4, the training phase of FRSVC has significant efficiency superiority but leaves relatively bigger N_{SV}. This might impose restrictions on the labeling efficiency, in practical use, even though it has the lowest time com-

plexity of $O(\ell N_{SV} + \tilde{m}N_c)$. Therefore, exploiting an effective way of reducing the size of SVs is essential. Actually, ℓN_{SV} is greater than $\tilde{m}N_c$ while $\ell > \tilde{m}$ and $N_{SV} > N_c$. That means a significant number of redundant SVs cuts down the labeling efficiency because they locate the same SEVs. Intuitively, we can set a threshold to remove most of them, e.g., fixing SV buffer size N_{SV} [40] or filtering out SVs with coefficient values lower than β_t [3]. Efficiencies of the both are not at the cost of accuracy, and their effectiveness can also be explained and supported by [1,2,8]. It is worth mentioning that an appropriate threshold should be set corresponding to specific applications. By focusing on presenting a flexible FRSVC with two fast phases, for universality, we have omitted finding the best time costs for FRSVC with reduced sizes of SVs and strongly recommend further application-aware investigations.

5. Related works

From Section 2, SVC includes the training phase and the labeling phase. 1) Inspired by SVMs, solving the dual problem is the key work for the training phase. Although it has similar form with the convex optimization problems of SVMs, its fundamental basis is the minimum enclosing ball (MEB) or minimal enclosing sphere (MES) problem, and it analyzes unlabeled data. Therefore, solutions motivated by large margin strategy with known labels for SVMs are not suitable for SVC in direct use. In the literature, the major researches prefer generic optimization algorithms (e.g. gradient descent, quasi-Newton, trust region methods, and so on) for this quadratic programming problem [1]. In spite of being designed for SVM, an noticeable work of budget-driven online learning method [40] which constructs decision function dynamically that might gives a fresh thought for SVC's solution. A few works adopt finding an alternative problem, e.g., sequential minimal optimization [41] and entropy-based algorithm [7] by introducing Jaynes maximum entropy principle to rewrite the dual problem (2). Besides, by measuring and introducing a position-based weight to each data sample, PSVC [23] constructs an alternative dual problem of (2) which shrinks the range of coefficient β. In PSVC, the penalty factor C no longer exists because of the position-based weight. Furthermore, FSSVC [3] drops C thoroughly in its dual problem which is constructed by particularly selected boundaries. In practical, most of them consume an unstable runtime close to or equal to $O(N^3)$, and a space complexity of $O(N^2)$ for the pre-computed kernel matrix is generally suggested. By considering FSSVC, here N is the actual size of data samples employed before solving the specific dual problem. Therefore, using a reduced N is direct for a better efficiency, and VCC [24] is another case. It uniformly samples a predefined ratio $\theta \in (0, 1]$ of data (i.e., $N_{tr} = \theta N$) to construct a train set for obtaining the support function in classic way. Although under specific strategies, few works try to change the optimizing methods. Additionally, the other methods related to reducing the working set and divide-and-conquer strategy were surveyed in [1].

For cluster labeling, generally, it results in huge time and storage requirements, such as the time complexity $O(mN^2)$ required by CG [5] who checks the line segments connecting the full N data samples' pairs. To accelerate the labeling speed, especially on large-scale data, some insightful strategies have been designed to replace CG. For instance, support vector graph (SVG)[5] uses SVs to reach $O(mN_{SV}^2)$. Intuitively, neighboring data samples are more likely to be grouped in a cluster rather than those locating far away. By considering this, Ref. [42] proposed proximity graph (PG) modeling methods for SVC. Based on different neighboring metrics, PGs for the Delaunay Diagram (DD), minimum spanning tree (MST) and k nearest neighboring (kNN) were suggested. They reduce the time complexity to $O(N \log N + mf(N))$, $O(N \log N + mkN)$ and $O(N \log N + mN)$, respectively. Here, $f(N)$ is a function of N for DD. Further, Ref[8]. found that the support function (3) can be considered as a dynamical system. According to the principle of support vector data description [43], in feature space, the locations from the inner points to the center of the hypersphere are closer than those from cluster boundaries or outliers. In a dynamical system, a SEV is a local optimum that can be reached by any optimization algorithm starting at any point. Then, a series of equilibrium based approaches were proposed one by one, e.g., R-CG [8], E-SVC [9,17,19,20], FSVC[2] and VCC [24]. R-CG linearly checks line segments connecting SEV pairs to determine the connectivity. It achieves improvement on efficiency with a slightly reduced accuracy. To fix this problem, E-SVC introduced a transition point (TS) between two SEVs. So, finding the connecting components depends on whether the distance from the corresponding TS to $\boldsymbol{\alpha}$ is less than R. However, R-CG and E-SVC were found with limited capability facing large scale or high dimensional data. Later, in [2], FSVC first use a k-means like algorithm as a preprocessing step to divide the data into N_b subsets. Next, N_b centers are employed to find the corresponding SEVs and the final task of labeling is achieved by R-CG. Recently, VCC [24] integrates advantages of E-SVC and FSVC in an optional way, i.e., choosing improved strategies of either E-SVC for accuracy (Mode I in Table 1) or FSVC for efficiency (Mode II in Table 1). Actually, in these approaches, SEV is a type of exemplar of a subset that distributed in an irregular region. By taking SEV as a component of cluster prototype, Ref. [18] added a definition of density centroid and developed a double centroids (DBC) labeling method where the membership is decided according to a weighted norm distance between $\mathbf{x}$ and each cluster. Furthermore, an insightful definition of convex hull was presented and used as the cluster prototype by [4] where CDCL was proposed. It frequently reaches a significantly reduced runtime without a loss of accuracy. Based on CDCL, FSSVC [3] is a faster and stable algorithm. It built a novel dual problem based on the selected critical patterns which saves both time and storage, and use an adaptive labeling strategy to integrate the advantages of CDCL and CCL [21,22]. CCL has an smart design to avoid sampling task. It constructs cone-shaped neighborhoods for SVs, and then observes the intersection situation among neighborhoods to generate the adjacency matrix of SVs. Two SVs belong to the same cluster if their cones intersect.

6. Conclusion

Towards making SVC consume less, we have proposed FRSVC with flexible solver for different conditions. The first core of the proposed methods lies in a novel and efficient DCD solver designed to solve the dual problem. Based on it, we can easily receive benefits from three aspects. Firstly, it has significant efficiency superiority, in comparison of the state-of-the-art methods, if the runtime platform (e.g., Cloud) is equipped with sufficient memory for the kernel matrix. Even if one platform cannot meet the storage requirement, the iterations of the DCD solver can also be separated on the basis of rows of kernel matrix. For that case, a parameter server might be required for synchronization. Secondly, if the platform can afford the whole data set rather than the kernel matrix, FRSVC is competent since it calculates the required kernel value in the iteration of DCD solver instead of storing the whole matrix. For practical use, thereafter, we can find a balance between time complexity and storage complexity on different platforms.

To avoid pricey computation, the second core of FRSVC is that it adopts the labeling phase being finished effectively with limited resources. So, a novel labeling method based on the discovered convex hulls is proposed. It finishes connectivity analysis by checking line segment connecting SV and SEV from two nearest neighboring convex hulls, respectively. Accompanied by a specific designed strategy of sample once for connection checking first, not

only a great reduction of average sample rate, but also significant improvements on accuracy are achieved.

As the aforementioned, FRSVC is suggested to be used by the data owner locally which effectively trades time for space. Even though their DCD solver runs fast, the employed prototype finding based labeling method (for SEVs) suffers from pricey iterations. Furthermore, to handle huge data sets, how to outsource them is expected to be tackled in future.

Acknowledgments

The authors would like to thank the Associate Editor and the anonymous reviewers for their constructive comments that greatly improved the quality of this manuscript. This work was supported by the grant from the National Natural Science Foundation of China under Grant No. 61303232 and 61540049, the Foundation of Henan Educational Committee under Grant No. 16A520025, the Foundation for University Key Teacher of Henan Province (No. 2016GGJS-141), and Outstanding Young Teacher Project of Xuchang University.

References

[1] H.N. Li, Y. Ping, Recent advances in support vector clustering: theory and applications, Int. J. Pattern Recognit. Artif. Intell. 29 (1) (2015) 1550002, doi:10.1142/S0218001415500020.
[2] K.-H. Jung, D. Lee, J. Lee, Fast support-based clustering method for large-scale problems, Pattern Recognit. 43 (5) (2010) 1975–1983.
[3] Y. Ping, Y. Chang, Y. Zhou, Y. Tian, Y. Yang, Z. Zhang, Fast and scalable support vector clustering for large-scale data analysis, Knowl. Inf. Syst. 43 (2) (2015) 281–310.
[4] Y. Ping, Y. Tian, Y. Zhou, Y. Yang, Convex decomposition based cluster labeling method for support vector clustering, J. Comput. Sci. Technol. 27 (2) (2012) 428–442.
[5] A. Ben-Hur, D. Horn, H.T. Siegelmann, V.N. Vapnik, Support vector clustering, J. Mach. Learn. Res. Dec (2) (2001) 125–137.
[6] G.W. Flake, S. Lawrence, Efficient svm regression training with smo, Mach. Learn. Spec. SVMs 46 (1–3) (2001) 271–290.
[7] C.H.F. Li, An improved algorithm for support vector clustering based on maximum entropy principle and kernel matrix, Expert Syst. Appl. 38 (7) (2011) 8138–8143.
[8] J. Lee, D. Lee, An improved cluster labeling method for support vector clustering, IEEE Trans. Pattern Anal. Mach. Intell. 27 (3) (2005) 461–464.
[9] D. Lee, J. Lee, Equilibrium-based support vector machine for semisupervised classification, IEEE Trans. Neural Netw. 18 (2) (2007) 578–583.
[10] K.-H. Jung, N. Kim, J. Lee, Dynamic pattern denoising method using multi-basin system with kernels, Pattern Recognit. 44 (8) (2011) 1698–1707.
[11] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S.S. Keerthi, S. Sundararajan, A dual coordinate descent method for large-scale linear svm, in: Proceedings of the 25th International Conference on Machine Learning (ICML '08), ACM, 2008, pp. 408–415.
[12] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, Cambridge, New York.
[13] A. Ben-Hur, D. Horn, H.T. Siegelmann, V. Vapnik, A support vector cluster method, in: Proc. the 15th International Conference on Pattern Recognition, 2000, pp. 724–727.
[14] J.H. Chiang, P.Y. Hao, A new kernel-based fuzzy clustering approach: support vector clustering with cell growing, IEEE Trans. Fuzzy Syst. 11 (4) (2003) 518–527.
[15] C.-H. Lee, H.-C. Yang, Construction of supervised and unsupervised learning systems for multilingual text categorization, Expert Syst. Appl. 2-1 (36) (2009) 2400–2410.
[16] D. Lee, K.-H. Jung, J. Lee, Constructing sparse kernel machines using attractors, IEEE Trans. Pattern Anal. Mach. Intell. 20 (4) (2009) 721–729.
[17] J. Lee, D. Lee, Dynamic characterization of cluster structures for robust and inductive support vector clustering, IEEE Trans Pattern Anal Mach Intell 28 (11) (2006) 1869–1874.
[18] Y. Ping, Y.J. Zhou, Y.X. Yang, A novel scheme for accelerating support vector clustering, Comput. Inf. 31 (3) (2011) 1001–1026.
[19] C.-H. Lee, H.-C. Yang, Construction of supervised and unsupervised learning systems for multilingual text categorization, Expert Syst. Appl. 2-1 (36) (2009) 2400–2410.
[20] D. Lee, K.-H. Jung, J. Lee, Constructing sparse kernel machines using attractors, IEEE Trans. Pattern Anal. Mach. Intell. 20 (4) (2009) 721–729.
[21] S.-H. Lee, K.M. Daniels, Cone cluster labeling for support vector clustering, in: Proceedings of 6th SIAM Conference on Data Mining, 2006, pp. 484–488.
[22] S.-H. Lee, K.M. Daniels, Gaussian Kernel Width Selection and Fast Cluster Labeling for Support Vector Clustering, Technical Report, No. 2005-009, Department of Computer Science, University of Massachusetts Lowell, USA, 2005.
[23] C.-D. Wang, J.-H. Lai, Position regularized support vector domain description, Pattern Recognit. 46 (3) (2013) 875–884.
[24] K. Kim, Y.Son, J. Lee, Voronoi cell-based clustering using a kernel support, IEEE Trans. Knowl. Data Eng. 27 (4) (2015) 1146–1156.
[25] R. Xu, D.C. Wunsch, Clustering, A John Wiley&Sons, Hoboken, New Jersey.
[26] F. Camastra, A. Verri, A novel kernel method for clustering, IEEE Trans. Pattern Anal. Mach. Intell. 27 (5) (2005) 801–805.
[27] H.-C. Kim, J. Lee, Clustering based on gaussian processes, Neural Comput. 19 (11) (2007) 3088–3107.
[28] A. Frank, A. Asuncion, UCI machine learning repository, 2010.
[29] Y. Ping, Y.J. Zhou, C. Xue, Y.X. Yang, Efficient representation of text with multiple perspectives, J. China Univ. Posts Telecommun. 19 (1) (2012) 101–111.
[30] M. Graven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, S. Slattery, Learning to extract symbolic knowledge form the world wide web, in: Proc. 15th Nat'l Conf. for Artificial Intelligence (AAAI'98), Madison, Wisconsin, USA, 1998, pp. 509–516.
[31] K. Lang, Newsweeder: learning to filter netnews, in: Proc. 12th Intl Conf. Machine Learning (ICML'95), Tahoe City, California, USA, 1995, pp. 331–339.
[32] D.D. Lewis, Reuters-21578 text categorization collection, 1997.
[33] W.R. Hersh, C. Buckley, T.J. Leone, D.H. Hickam, Ohsumed: an interactive retrieval evaluation and new large test collection for research, in: Proceedings of 17th Annual ACM SIGIR Conference, 1994, pp. 192–201.
[34] UNIBS, The unibs anonymized 2009 internet traces, Mar 18, 2010.
[35] J.F. Peng, Y.J. Zhou, C. Wang, Y.X. Yang, Y. Ping, Early tcp traffic classification, J. Appl. Sci. Electr. Inf. Eng. 29 (1) (2011) 73–77.
[36] S. Garcia, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, J. Mach. Learn. Res. 9 (Dec) (2008) 2677–2694.
[37] D. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, Chapman & Hall/CRC.
[38] G. Bergmann, G. Hommel, Improvements of general multiple test procedures for redundant systems of hypotheses, in: Multiple Hypotheses Testing, vol. 70, 1988, pp. 100–115.
[39] M. Wu, B. Scholkopf, A local learning approach for clustering, in: Advances in Neural Information Processing Systems (NIPS 2007), Vancouver, B.C., Canada, 19, 2007, pp. 1529–1536.
[40] Y. Qian, H. Yuan, M. Gong, Budget-driven big data classification, Lect. Notes Comput. Sci. 9091 (2015) 71–83.
[41] J.C. Platt, Fast training of support vector machines using sequential minimal optimization, in: Advances in Kernel Methods: Support Vector Learning (MIT Press, 1999), 1999, pp. 185–208.
[42] V. Estivill-Castro, I. Lee, A.T. Murray, Criteria on proximity graphs for boundary extraction and spatial clustering, in: Proceedings of the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'01), vol. 2035, 2001, pp. 348–357.
[43] D.M. Tax, R.P. Duin, Support vector data description, Mach. Learn. 54 (1) (2004) 45–66.

Yuan Ping received the PhD degree from Beijing University of Posts and Telecommunications, China, in 2012. He was a research assistant in University of Alberta from September 2014 to August 2015. He is currently an Associate Professor, and the Director of Network and Information Security Lab of Xuchang University. His research interests are in machine learning, data mining, pattern recognition, information security, and operating system.

Yingjie Tian received the PhD in Management Science and Engineering from China Agricultural University in 2005. He is currently a Full Professor, and Vice Director of Research Center on Fictitious Economy & Data Science, Chinese Academy of Sciences. He has published 4 books about SVMs, one of which has been cited over 1000 times. His research interests include SVMs, optimization theory and applications, data mining, intelligent knowledge management.

Chun Guo received B.Sc. and M.Sc in July 2008 and July 2011 from GuiZhou University, respectively, and received Ph.D. in information security from Beijing University of Posts and Telecommunications in July 2014. He is currently a lecturer in the College of Computer Science and Technology, GuiZhou University. His research interests include data mining, intrusion detection and intrusion defense.

Baocang Wang is a professor in the School of Telecommunications Engineering, Xidian University. He received his PhD degree in cryptography from Xidian University in 2006, and received his MS and BS degrees in mathematics from Xidian University in 2004 and 2001, respectively. His main research interests include public key cryptography, wireless network security and data mining.

Yuehua Yang received her doctor's degree from Beijing University of Posts and Telecommunications in 2013. She is currently a lecturer in School of Information Engineering at Xuchang University. Her research interests include intelligent information system and knowledge base.