Journal of Universal Computer Science, vol. 15, no. 4 (2009), 705-721
submitted: 30/6/08, accepted: 27/2/09, appeared: 28/2/09 © J.UCS

An Efficient Data Preprocessing Procedure for Support Vector Clustering

Jeen-Shing Wang
(Department of Electrical Engineering, National Cheng Kung University
Tainan City 701, Taiwan, R.O.C.
jeenshin@mail.ncku.edu.tw)

Jen-Chieh Chiang
(Department of Electrical Engineering, National Cheng Kung University
Tainan City 701, Taiwan, R.O.C.
n2894108@mail.ncku.edu.tw)

Abstract: This paper presents an efficient data preprocessing procedure for the support of vector clustering (SVC) to reduce the size of a training dataset. Solving the optimization problem and labeling the data points with cluster labels are time-consuming in the SVC training procedure. This makes using SVC to process large datasets inefficient. We proposed a data preprocessing procedure to solve the problem. The procedure contains a shared nearest neighbor (SNN) algorithm, and utilizes the concept of unit vectors for eliminating insignificant data points from the dataset. Computer simulations have been conducted on artificial and benchmark datasets to demonstrate the effectiveness of the proposed method.

Keywords: Support Vector Clustering, Shared Nearest Neighbors, Noise Elimination
Categories: I.2.6, I.5.0, I.5.1, I.5.3

1 Introduction

Due to the rapid growth of computer and information technologies, databases become larger and larger, which increases the need of more efficient and effective analytical tools to analyze and retrieve useful information/knowledge from databases. Clustering algorithms are useful for discovering groups and distributions in large databases and have been widely adopted in diverse scientific fields and commercial sectors. Clustering algorithms partition a dataset into clusters or classes, where similar data are grouped into the same cluster and dissimilar data are grouped into different clusters. In recent years, a number of clustering algorithms have been proposed [Ankerst et al. 1999, Ester et al. 1996, Guha et al. 1998, Guha et al. 1999, Zhang et al. 1996] for dealing with large databases. These algorithms are capable of finding clusters with different shapes, sizes, densities, and even in the presence of noise and outliers in datasets. Although these algorithms can handle clusters with different shapes, they still cannot produce arbitrary cluster boundaries to adequately capture or represent the characteristics of clusters in the dataset. Support vector clustering (SVC) [Ben-Hur et al. 2000, Cortes and Vapnik 1995] can overcome the limitation of these clustering algorithms. The SVC algorithm, first proposed by Ben-Hur *et al.*, identifies the cluster contours with arbitrary geometric representations, and automatically determines the number of clusters for a given dataset by a unified framework. The SVC algorithm has been widely researched in both theoretical developments and practical applications due to its outstanding features [Ben-Hur et al. 2000, Cortes and Vapnik 1995]. In the SVC algorithm, data points are mapped from the data space to a high dimensional feature space using Gaussian kernels. The objective of the SVC algorithm is to look for the smallest sphere that encloses the images of data points in the feature space. This sphere is then mapped back to the data space, where a number of contours which enclose the data points are formed. These contours are interpreted as cluster boundaries. In general, the SVC algorithm involves three main steps [Saketha Nath and Shevade 2006]: 1) finding the hyper-sphere by solving the Wolfe dual optimization problem, 2) identifying the clusters by labeling the data points with cluster labels, and 3) searching a satisfactory clustering outcome by tuning kernel parameters.

In our previous research work [Wang and Chiang 2008a, Wang and Chiang 2008b], we have developed an effective parameter search algorithm to automatically search suitable parameters for the SVC algorithm. However, there is a common agreement in SVC research community—solving the optimization problem and labeling the data points with cluster labels are time-consuming in the SVC training procedure. The above limitations make the SVC algorithm inapplicable for large datasets. From our review of literature, we found that many research efforts have been conducted to improve the effectiveness of cluster labeling. Because the computation of cluster labeling is considerably expensive, many researchers have engaged in reducing time complexity of this aspect. Yang *et al.* [Yang et al. 2002] used proximity graphs to model the proximity structure of datasets. Their approach constructed appropriate proximity graphs to model the proximity and adjacency. After the SVC training process, they employed cutoff criteria to estimate the edges of a proximity graph. This method avoids redundant checks in a complete graph, and also avoids the loss of neighborhood information as it can occur when only estimating the adjacencies of support vectors. Lee and Lee [Lee and Lee 2005] created a new cluster labeling method based on some invariant topological properties of a trained kernel radius function. The method they proposed consisted of two phases. The first phase was to decompose a given data set into a small number of disjoint groups where each group was represented by its candidate point and all of its member points belong to the same cluster. The second phase was then to label the candidate points. Nath and Shevade [Saketha Nath and Shevade 2006] presented a novel approach that increases the efficiency of the SVC scheme. The geometry presented in the clustering problem was exploited to reduce the training data size. Their experiments showed that the preprocessing procedure drastically decreased the run-time of the cluster algorithm. However, different pre-specified parameters could produce totally different clustering results.

Based on the above discussion, we proposed an efficient data preprocessing procedure to accelerate the training of SVC without significantly altering the final cluster configuration. The proposed procedure ameliorates the drawbacks of the SVC algorithm for dealing with large datasets. The preprocessing procedure utilizes a shared nearest neighbor (SNN) algorithm for eliminating the noise points, and the concept of unit vectors for removing the core points from the dataset. Since the size of the dataset is reduced, the computational burden for solving the optimization problems as well as cluster labeling can be greatly decreased.

The organization of this paper is as follows. The overview of the SVC algorithm is provided in Section 2. In Section 3, the proposed data preprocessing procedure for the SVC algorithm is introduced in detail. The simulation results on artificial and benchmark datasets are presented in Section 4. Finally, conclusions are given in Section 5.

2 Support Vector Clustering

The mathematical formulation of the SVC algorithm is summarized as follows. Assume a dataset containing N points $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}, \mathbf{x}_i \in \mathbb{R}^d$, where d is the dimension of the data space. A nonlinear mapping function $\mathbf{\Phi}$ is used to map the data set into a high-dimensional feature space such that the radius of the sphere, denoted by R, enclosing all the data points is as small as possible. Such an objective can be formulated by the following optimization problem:

$$\begin{aligned} &\min\ R^2 + C\sum_j \xi_j \\ &\text{subject to } \left\|\Phi(\mathbf{x}_j) - \mathbf{a}\right\|^2 \leq R^2 + \xi_j \quad \forall j \end{aligned} \qquad (1)$$

where $\|\cdot\|$ is the Euclidean norm, $\mathbf{a}$ is the center of the sphere, ξ_j are slack variables that loosen the constraints to allow some data points lying outside the sphere, C is a constant, and $C\sum\xi_j$ is a penalty term. To solve the optimization problem in (1), it is convenient to introduce the Lagrangian function:

$$L(R, \mathbf{a}, \xi_j, \beta_j, \mu_j) = R^2 - \sum_j (R^2 + \xi_j - \left\|\Phi(\mathbf{x}_j) - \mathbf{a}\right\|^2)\beta_j - \sum \xi_j \mu_j + C\sum \xi_j, \qquad (2)$$

where $\beta_j \geq 0$ and $\mu_j \geq 0$ are the Lagrange multipliers. With (2), we can derive the following conditions by the Lagrange theorem and the Karush-Kuhn-Tucker (KKT) complementarity [Cortes and Vapnik 1995].

$$\xi_j \mu_j = 0, \qquad (3)$$

$$\left(R^2 + \xi_j - \left\|\Phi(\mathbf{x}_j) - \mathbf{a}\right\|^2\right)\beta_j = 0.$$

(4)

According to (3) and (4), we can classify each data point into 1) an internal point, 2) an external point, and 3) a boundary point in the feature space. Point $\mathbf{x}_j$ is classified as an internal point if $\beta_j = 0$. When $0 < \beta_j < C$, the data point $\mathbf{x}_j$, is denoted as a support vector (SV). SVs lying on the surface of the feature-space sphere are the so-called boundary points. These SVs can be used to describe the cluster contour in the input space. When $\beta_j = C$, the data points located outside the feature space are defined as the external points or bounded support vectors (BSVs).

Using the above conditions, (1) can be turned into the Wolfe dual optimization problem with only variables β_j:

$$\max \sum_j \Phi(\mathbf{x}_j)^2 \beta_j - \sum_{i,j} \beta_i \beta_j \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$
$$\text{subject to } 0 \le \beta_j \le C \text{ and } \sum_j \beta_j = 1, \forall j, \tag{5}$$

where the dot product of ($\Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j)$) represents the Mercer kernel $K(\mathbf{x}_i, \mathbf{x}_j)$. Here, we select Gaussian functions as kernels, i.e., $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-q \lVert \mathbf{x}_i - \mathbf{x}_j \rVert^2)$. For any point $\mathbf{x}$ in the data space, the distance of its image in the feature space from the center of the sphere is given by

$$R^2(\mathbf{x}) = \lVert \Phi(\mathbf{x}) - \mathbf{a} \rVert^2 = K(\mathbf{x}, \mathbf{x}) - 2\sum_j \beta_j K(\mathbf{x}_j, \mathbf{x}) + \sum_{i,j} \beta_i \beta_j K(\mathbf{x}_i, \mathbf{x}_j). \tag{6}$$

The radius R of the sphere can be obtained by

$$R = \{R(\mathbf{x}_i) \mid \mathbf{x}_i \text{ is a support vector}\}. \tag{7}$$

In practice, the average of the above set is used as the radius R. The SVs, BSVs, and the other points are located on the cluster boundaries, the outside of the boundaries, and the inside of the boundaries, respectively. From the above discussion, we found that there are two important user-specified parameters: q and C. The value of q governs the number of clusters and the smoothness/tightness of the cluster boundaries as well, while the value of C determines the existence of outliers during the clustering process.

The above SVC training procedure determines only the cluster contours of the data set. The cluster description itself does not differentiate points that belong to different clusters. As noted in [Ben-Hur et al. 2000, Cortes and Vapnik 1995], if there are two data points, $\mathbf{x}_i$ and $\mathbf{x}_j$, that belong to the same cluster in the input space, one can check if the line segment between them always travels within the high dimensional sphere. Checking the line segment is implemented by sampling a number of points on the segment (usually 10-20 points). Two data points, $\mathbf{x}_i$ and $\mathbf{x}_j$, satisfying the above condition are defined as connected components. An adjacency matrix $\boldsymbol{A}$ is defined to identify the connected components of a cluster. We define the components of $\boldsymbol{A}$, a_{ij}, between pairs of points $\mathbf{x}_i$ and $\mathbf{x}_j$:

$$a_{ij} = \begin{cases} 1, & \text{if all } \mathbf{y} \text{ on the line segment connecting } \mathbf{x}_i \text{ and } \mathbf{x}_j,\ R(\mathbf{y}) \le R. \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

The values of a_{ij} can be obtained by sampling a number of points from the line segment connecting $\mathbf{x}_i$ and $\mathbf{x}_j$. In the matrix $\boldsymbol{A}$, if $a_{ij} = 1$ that means $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same cluster; otherwise, they are in different clusters. In general, the cluster labeling step that checks the connectivity for each pair of samples is more time-consuming than the SVC training step. The time complexity of this procedure is $O(lN^2)$, where l is the number of samples on the line segment.

3 An Efficient Data Preprocessing Procedure for SVC

Solving the optimization problem and labeling the data points with cluster labels are time-consuming in the SVC training procedure. This makes using the SVC algorithm to process large datasets inefficient. Thus, how to exclude redundant data points from a dataset is an important issue for minimizing the time spent in solving the optimization problem of the SVC algorithm. Our research challenge in this topic is how to identify insignificant data points so that the removal of these data points does not significantly alter the final cluster configuration. Our idea is to eliminate insignificant data points, such as noise and core points, from the training datasets, and use the remaining data points to do the SVC analysis. Due to the size reduction of the

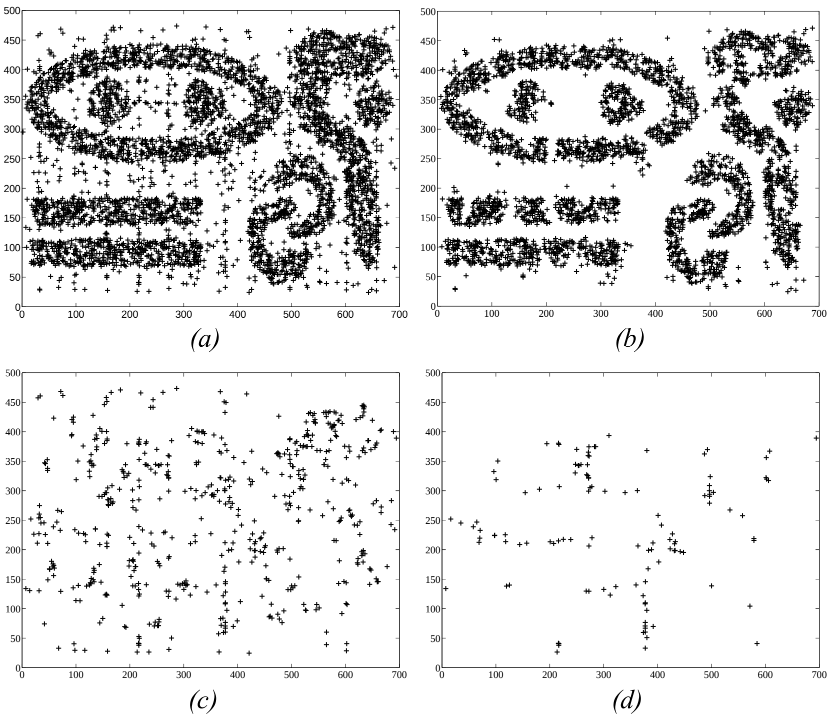


Figure 1: A 2D dataset example for using SNN. (a) The original dataset. (b) The distribution of highly similar data points. (c) The distribution of medium similar data points. (d) The distribution of low similar data points.

training datasets, the computational effort for solving the optimization problem can be greatly decreased. To fulfil the idea, we first explore the shared nearest neighbor (SNN) algorithm [Ertöz et al. 2003, Jarvis and Patrick 1973] to eliminate noise points. Subsequently, the concept of unit vectors [Saketha Nath and Shevade 2006] is employed to reduce the core points of clusters and to retain the data points near the cluster boundaries. Based on these two methods, we developed an efficient data preprocessing procedure for SVC to reduce the size of the training datasets without altering the cluster configuration of the datasets.

3.1 Elimination of Noise Points by A Shared Nearest Neighbor Algorithm

A shared nearest neighbor (SNN) algorithm proposed by Jarvis and Patrick [Jarvis and Patrick 1973] first finds the nearest neighbors of each data point, and then computes the similarity between pairs of points in terms of how many nearest neighbors each pair of the data points shares. The SNN algorithm can help us to eliminate noise and outliers, and to identify core points that are the representative points from the regions with relatively high densities. The representative points are then further processed by the concept of unit vectors to remove the insignificant points from the core points. Ideally, the remaining points are the boundary points that can depict the cluster contours of the original datasets.

To eliminate noise points and outliers, the SNN algorithm first obtains a similarity matrix whose components are defined as the similarity measure between a pair of points [Ertöz et al. 2003, Jarvis and Patrick 1973]. The similarity measure between a pair of points is defined as follows. First, a link is created between a pair of points, r and s, if and only if r and s have each other in the list of their k_1 nearest neighbors, where k_1 is a user pre-specified parameter. The strength of a link between two points is expressed by the number of nearest neighbors that are shared by the two points. Specifically, if r and s are the two points, the strength of the link between r and s, their similarity is defined as:

$$similarity(r, s) = \text{size}\,(NN(r) \cap NN(s)). \tag{9}$$

where $NN(r)$ and $NN(s)$ are the nearest neighbor lists of r and s, respectively.
Figure 1 illustrates the results of the original dataset after removing noise points and outliers and identifying core points by using SNN. The original dataset contains 5000 points. We set the number of nearest neighbors in the list of r or s, k_1, equals to 20. If the value of $similarity(r, s)$ is greater than or equal to α, we define that the points r and s are close (or similar) to each other. In this example, we set $\alpha = 10$. Figure 1(b) shows the data distribution of the points whose numbers of commonly shared nearest neighbors (CSNN) are more than 15 points. In this case, we say that the points are highly similar to each other. Figure 1(c) shows the distribution of the points whose CSNN numbers are greater than 10 but less than 14 points. We define that the points are medium similar to each other. Likewise, Figure 1(d) shows the distribution of low similar data points whose CSNN numbers are less than 10 points. In this study, the highly similar points are defined as core points and the low similar data points are noise points and outliers.
The steps of the SNN algorithm [Ertöz et al. 2003] for eliminating the noise points are as follows:
Step 1: Initialization. Set k_1 and calculate the similarity matrix.
Step 2: Closeness computation. Set α. Here, we set the value α as the average strength of all data points:

$$\alpha = \frac{\sum_{i=1}^{m}\sum_{j=1}^{k_1} similarity(\mathbf{x}_i, \mathbf{x}_j)}{m \times k_1}, \tag{10}$$

where m is the total number of data points. If the strength, $similarity(r, s)$, between the two points is greater than α, these two points are close to each other.
Step 3: Removal of noise points and outliers. First, we define a threshold δ that is used to define low-similar data points.

$$\delta = S_{mean} - S_{std}, \tag{11}$$

where

$$S_{mean} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{k_1} Count_{i,j}}{m}, \tag{12}$$

and

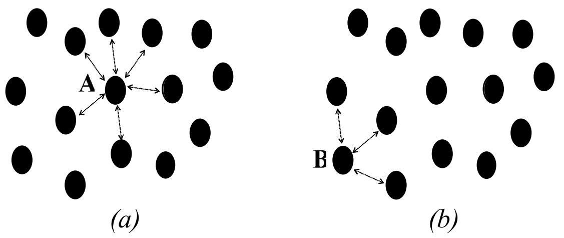


Figure 2: (a) Point A is a core point. (b) Point B is a boundary point.

$$S_{std} = \left\{ \frac{1}{m-1} \sum_{i=1}^{m} \left[\left(\sum_{j=1}^{k_1} Count_{i,j} \right) - S_{mean} \right]^2 \right\}^{\frac{1}{2}}. \tag{13}$$

$Count_{i,j}$ in (12) and (13) is defined as:

$$Count_{i,j} = \begin{cases} 1, \text{ if } similarity(\mathbf{x}_i, \mathbf{x}_j) \geq \alpha \\ 0, \text{ otherwise} \end{cases}, \tag{14}$$

where $1 \leq i \leq m$ and $1 \leq j \leq k_1$. For a specific point $\mathbf{x}_i$, if the strength of $\mathbf{x}_i$, $\sum_{j=1}^{k_1} Count_{i,j} \leq \delta$, $\mathbf{x}_i$ is defined as a noise point or a outlier and is removed from the dataset.

Step 4: End of the SNN algorithm. After eliminating the noise points, the SNN algorithm is completed.

3.2 Elimination of Core Points by the Concept of Unit Vectors

After the SNN algorithm is performed, most of noise points or outliers are removed from the datasets. We hope that the proposed data preprocessing procedure does not significantly alter the final cluster configurations but can save the computational time of SVC. Therefore, we need to eliminate non-support vector data points, such as core points. To achieve the objective, we further propose a method based on the concept of unit vectors [Saketha Nath and Shevade 2006] to eliminate the core points and retain the representative data points that are near the cluster boundaries. Figure 2 shows the difference between a core point and a boundary point. Figure 2(a) indicates point *A* is a core point that has neighboring points from all directions, while in Fig. 2(b), point *B* is a boundary point that has neighbors from only certain directions. For a data point $\mathbf{x}_p$, the summation of the unit vectors drawn from $\mathbf{x}_p$ to its k_2 nearest neighbors is calculated and defined as λ_p:

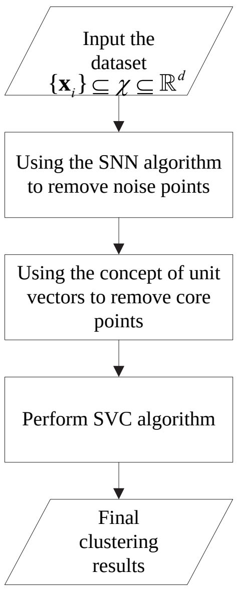

Input the dataset $\{\mathbf{x}_i\} \subseteq \chi \subseteq \mathbb{R}^d$
Using the SNN algorithm to remove noise points
Using the concept of unit vectors to remove core points
Perform SVC algorithm
Final clustering results

Figure 3: The flowchart of the proposed efficient data preprocessing procedure for SVC.

$$\lambda_p = \sum_{q=1}^{k_2} \frac{\mathbf{x}_p - \mathbf{x}_q}{\left\| \mathbf{x}_p - \mathbf{x}_q \right\|}, \tag{15}$$

where p = 1, 2, …, h, and h is the total number of the remaining data points after using the SNN algorithm to remove noise points. The smaller the value of λ_p is, the higher the possibility of $\mathbf{x}_p$ being a core point is. This is because the possibility of $\mathbf{x}_p$ having neighboring points from all directions is higher.

To eliminate core points by the concept of unit vectors, we define θ_1 as the average value of λ_p, p = 1, 2, …, h, by:

$$\theta_1 = \frac{1}{h} \sum_{p=1}^{h} \lambda_p . \tag{16}$$

If λ_p is smaller than θ_1, $\mathbf{x}_p$ is defined as a core point and is removed from the dataset. We further extend the above idea to remove noise points. We consider the points located far away from core points as noise points. We define θ_2 as the distance for distinguishing noise points from all the remaining data points. The distance is expressed by

$$\theta_2 = \theta_1 + \left[\frac{1}{h-1} \sum_{p=1}^{h} \left(\lambda_p - \theta_1 \right)^2 \right]^{\frac{1}{2}}, \tag{17}$$

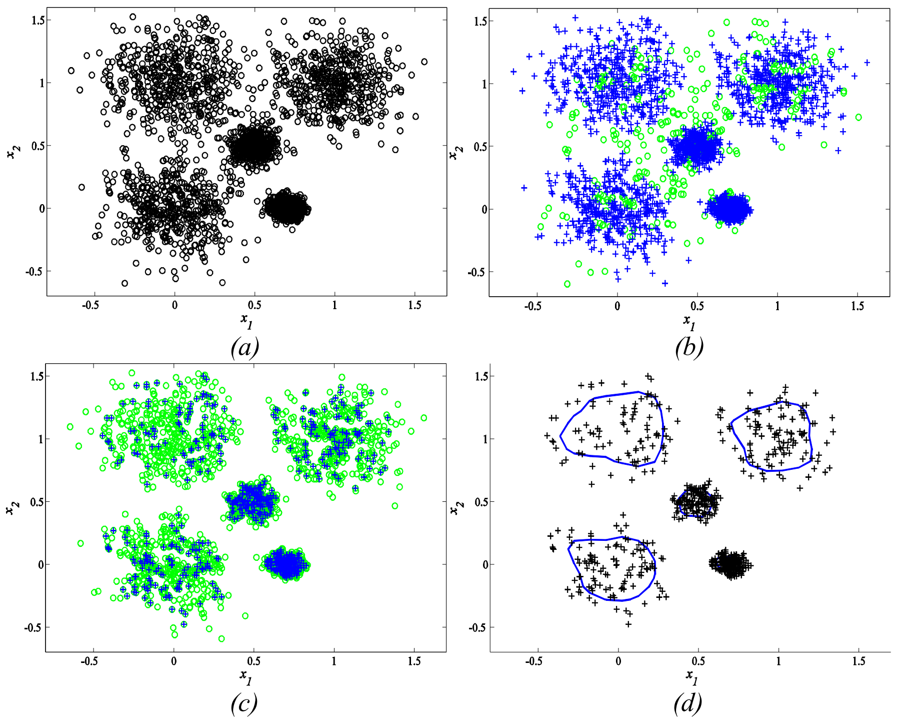


Figure 4: (a) The original artificial dataset. (b) The noise points marked with circles are eliminated from the dataset. (c) The distributions of core and representative points. The core points marked with circles are eliminated from the dataset. (d) The final clustering result obtained by SVC. The contours represent cluster boundaries.

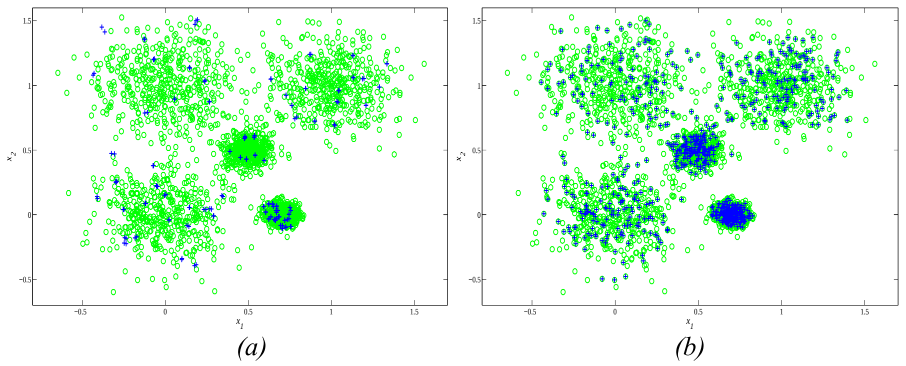
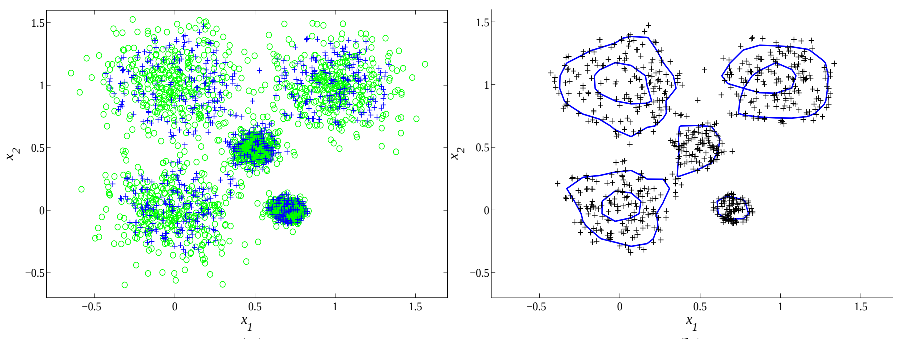


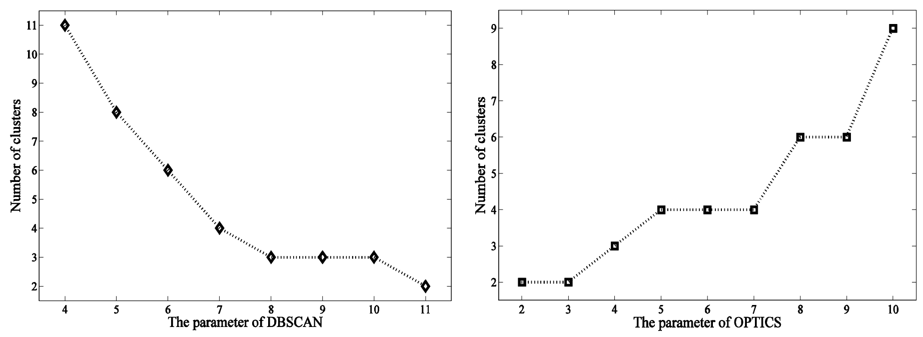
Figure 5: (a) More data points are removed if the value of α is large. (b) More noise points are retained if the value of α is small.

which equals to θ_1 plus a standard deviation of λ_p. For a data point $\mathbf{x}_p$, if λ_p is bigger than θ_2, it is classified as a noise point and is removed from the dataset. We summarize the above procedure for removing core and noise points as follows. First,



(a) *(b)*

Figure 6: (a) The core and noise points marked with circles can be removed from the artificial dataset. (b) The final clustering result obtained by SVC. Cluster boundaries are denoted by the contours.



(a) *(b)*

Figure 7: (a) The clustering results of DBSCAN. (b) The clustering results of OPTICS.

for each of the remaining data points, say $\mathbf{x}_i$ ($i = 1, \ldots, h$), we calculate the summation of the unit vectors from its k_2 nearest neighbors, denoted as λ_i. We then compute θ_1 and θ_2 according to (16) and (17). We classify each data point by the following conditions:

1. If $\lambda_i \leq \theta_1$, $\mathbf{x}_i$ is classified as a core point.
2. If $\lambda_i \geq \theta_2$, $\mathbf{x}_i$ is classified as a noise point.
3. Otherwise, $\mathbf{x}_i$ is a representative point of the dataset and will be used in the SVC training procedure.

The elimination of insignificant data points, e.g., noise and core points, from the dataset will not alter the final cluster configuration of the SVC algorithm but greatly improve its efficiency. There is because the computational complexity of solving the optimization problem and labeling the data points with cluster labels in the SVC algorithm can significantly decreased by reducing of the size of the training dataset. The overall time complexity of our proposed method is $O(N\log N)$, where N is the number of points. Figure 3 shows the flowchart of the proposed data preprocessing procedure for SVC.

4 Simulation Results

The effectiveness of the proposed data preprocessing procedure for the SVC algorithm has been validated through extensive computer simulations of different examples. We compared our proposed approach with the HRE method [Saketha Nath and Shevade 2006]. The HRE method is an efficient clustering scheme using support vector methods but requires users to pre-specified 8 parameters. We also compared our proposed approach with two well-known density-based methods, Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [Daszykowski et al. 2001] and Ordering Points to Identify the Clustering Structure (OPTICS) [Daszykowski et al. 2002] that are good at dealing with large datasets. DBSCAN is to create cluster with minimum size and density. This algorithm grows regions with sufficiently high density into clusters and discovers clusters of arbitrary shape in spatial databases with noise. Here, we adopt a modified DBSCAN method [Daszykowski et al. 2001] for comparing with our approach. The modified DBSCAN method only requires one user-specified parameter while the original DBSCAN [Ester et al. 1996] has two parameters to be specified. OPTICS is a density-based method that computes an augmented clustering ordering for automatic and interactive cluster analysis. The ordering represents the density-based clustering structure of the dataset. We provide

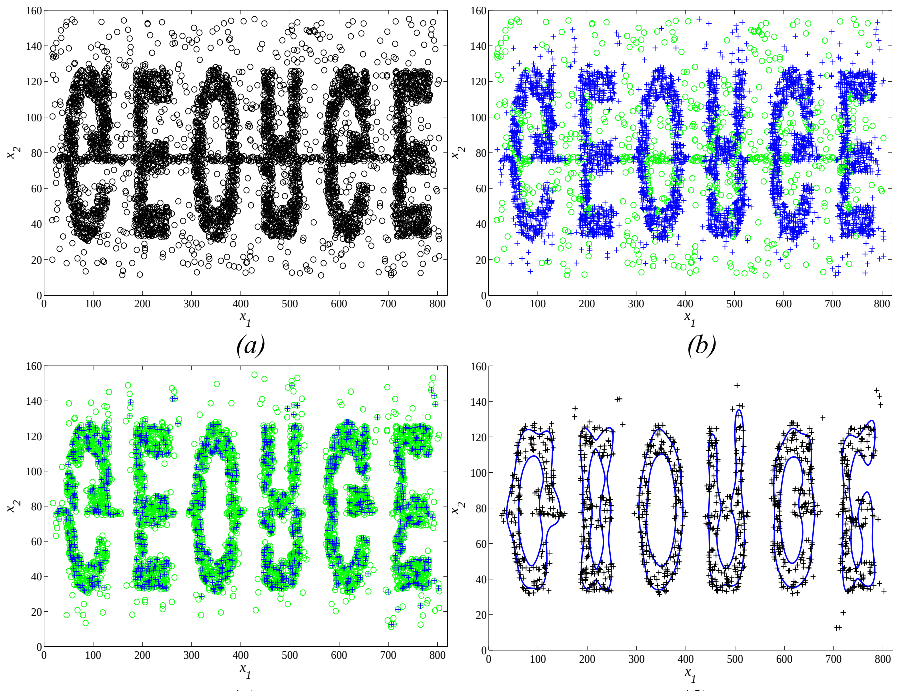


Figure 8: (a) The original benchmark dataset I. (b) The noise points marked with circles are eliminated. (c) The distributions of core and representative points. The core points marked with circles are eliminated from the dataset. (d) The final clustering result obtained by SVC.

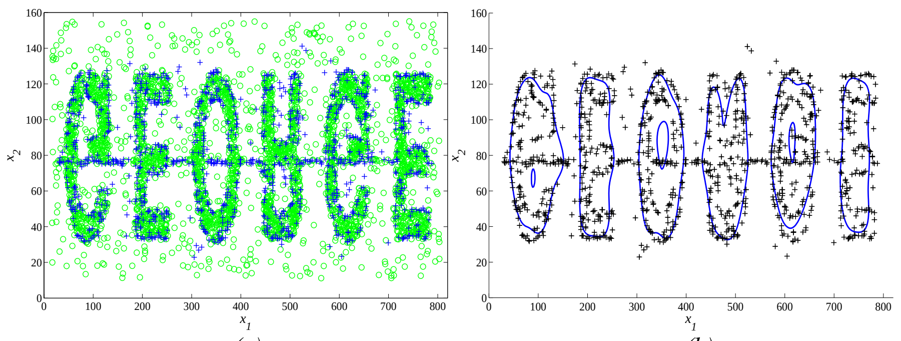
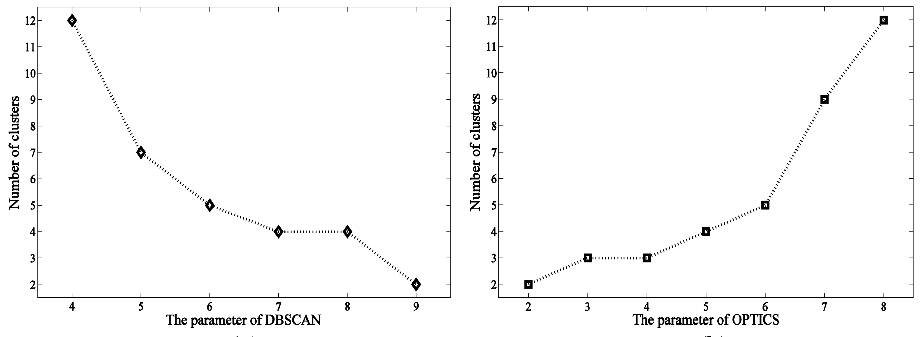

(a) (b)

Figure 9: (a) The core and noise points marked with circles can be removed from the benchmark dataset I. (b) The final clustering result obtained by SVC.



(a) (b)

Figure 10: (a) The clustering results of DBSCAN. (b) The clustering results of OPTICS.

three examples that contain the artificial, benchmark datasets [Karypis et al. 1999] and the Wisconsin breast cancer dataset [Black et al. 1998]. These 2-dimensional datasets contain 3000 to 5000 points with arbitrary shapes of clusters, various densities, and much noise.

4.1 Artificial Dataset

The artificial dataset consists of 3000 data points in a two-dimensional space. There are five different sizes of clusters in the dataset. We set k_1 = 40 and k_2 = 10 in this example and obtained α = 23.29, δ = 13.06, and λ_p = 3.77 from (10), (11), and (15), respectively. Figure 4(a) shows the distribution of the original artificial dataset. We used the SNN algorithm to eliminate the noise points and the result is shown in Fig. 4(b). In this step, a total of 529 data points were eliminated. Next, we used the concept of unit vectors to eliminate the core points and 712 data points were retained. The result is shown in Fig. 4(c). The execution time of our proposed method was 1.88 sec. Finally, we set q = 20 and C = 0.01 to obtain the final clustering results using the SVC algorithm. The clustering result is illustrated in Fig. 4(d). We performed more experiments with different α for comparison. In Fig 5(a), We set α = 38. Too many

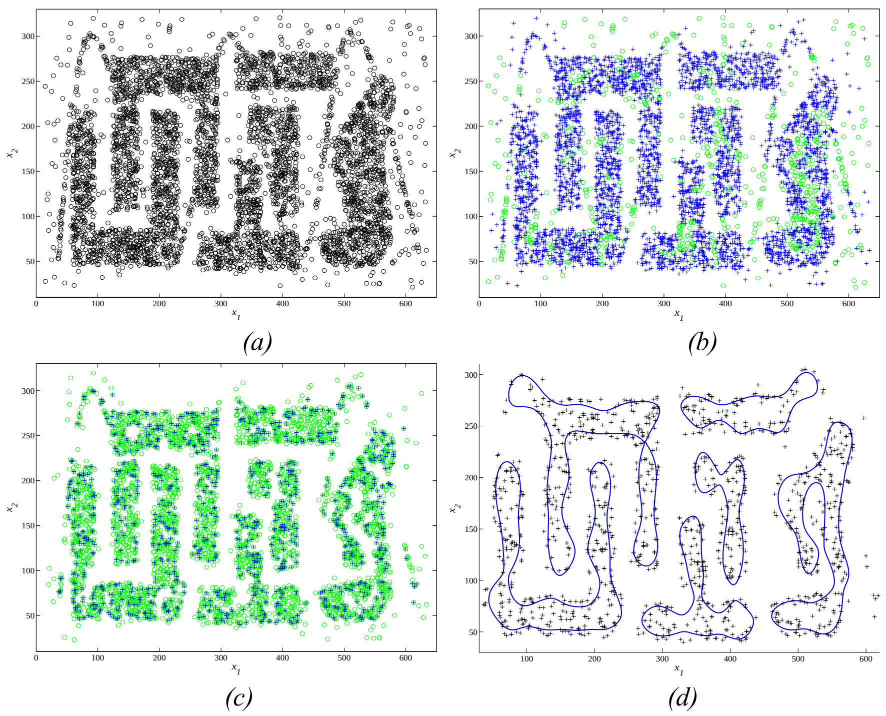


Figure 11: (a) The original benchmark dataset II. (b) The noise points marked with circles are eliminated from the dataset. (c) The distributions of core and representative points. The core points marked with circles are eliminated from the dataset. (d) The final clustering result obtained by SVC.

data points were removed, which resulted in an unsatisfactory clustering outcome. In Fig 5(b), We set $\alpha = 5$. Most data points including noise data were retained in the dataset, which increased the computational time of SVC. Thus, it is important to choose a suitable parameter α. The result of the data preprocessing step using the HRE method is shown in Fig. 6(a). The execution time of the HRE method was 2.62 sec. 712 data points were retained in the dataset. By setting $q = 20$ and $C = 0.01$, we obtained the final clustering result shown in Fig. 6(b). The hollow contours were not an ideal clustering outcome. The artificial set was also tested by two density-based clustering algorithms—DBSCAN [Daszykowski et al. 2001] and OPTICS [Daszykowski et al. 2002]. Figure 7 illustrates the clustering results of DBSCAN and OPTICS. The cluster number determined by our proposed method equals five, but DBSCAN and OPTICS cannot find the correct cluster number. Obviously, this example confirms that our proposed method is more accurate and efficient than HRE, DBSCAN and OPTICS.

4.2 Benchmark Datasets

1) Benchmark Dataset I

The benchmark dataset I consists of 5000 data points in a two-dimensional space with a large amount of noise points. There are six clusters in this dataset and the clusters are not linearly separable. We set k_1 = 55 and k_2 = 10 in this example and obtained α = 35.87, δ = 18.37, and λ_p = 3.91 from (10), (11), and (15), respectively. The original dataset is shown in Fig. 8(a). Figure 8(b) indicates the result obtained by the SNN algorithm with the removal of 788 noise points from the original dataset. There were 1176 data points retained after using the concept of unit vectors to eliminate the core points. The result is shown in Fig. 8(c). The execution time of our proposed method was 43.87 sec. Figure 8(d) illustrates the clustering results obtained by the SVC algorithm with q = 0.002 and C =0.01. The result obtained by the HRE method is shown in Fig. 9(a). There were 1176 data points retained in the dataset. The execution time of the HRE method was 57.62 sec. By setting q = 0.002 and C = 0.01, we obtained the final cluster result that is shown in Fig. 9(b). Because some of the noise points between characters were not removed completely, the clustering outcome was not as good as that of our approach. Our proposed method can identify the correct cluster number that equals six, but DBSCAN and OPTICS cannot obtain the correct number with different selections of the parameters. Figure 10(a) and (b) show the numbers of clusters vs. the values of the parameters for DBSCAN and OPTICS, respectively.

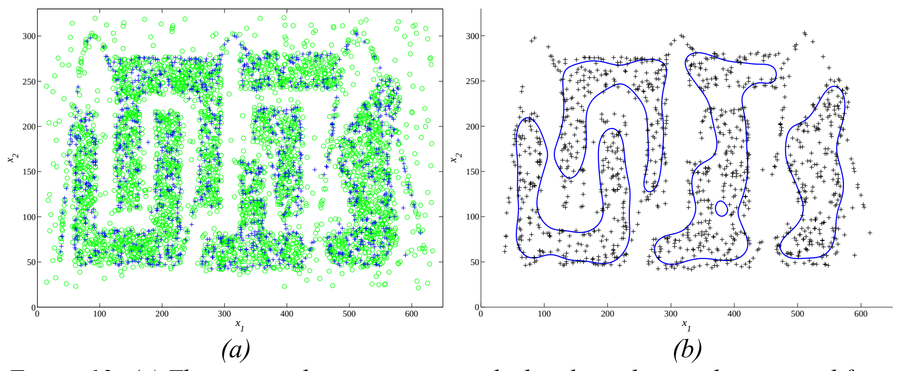
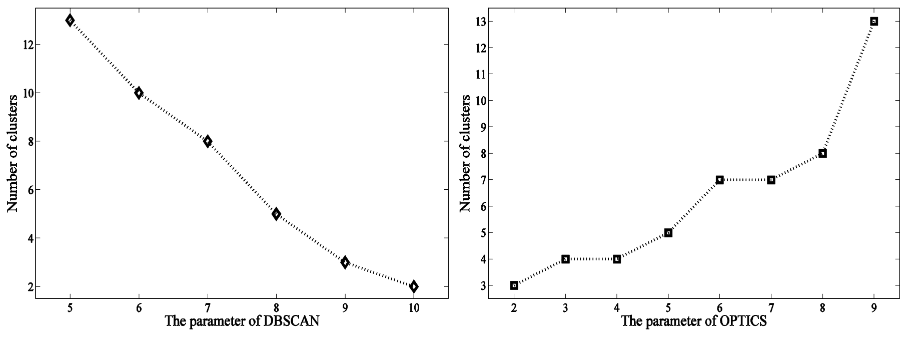


Figure 12: (a) The core and noise points marked with circles can be removed from the benchmark dataset II. (b) The final clustering result obtained by SVC.

2) Benchmark Dataset II

The benchmark dataset II also consists of 5000 data points and six clusters in a two-dimensional space with a large amount of noise points. We set k_1 = 55 and k_2 = 10 in this example and obtained α = 34.23, δ = 17.85, and λ_p = 3.72 from (10), (11), and (15), respectively. Figure 11(a) illustrates the original dataset. Figure 11(b) indicates the result obtained by the SNN algorithm that eliminates 773 noise points. Finally, there were 1201 data points retained after using the concept of unit vectors to eliminate the core points and the result is illustrated in Fig. 11(c). The execution time of our proposed method was 44.94 sec. Figure 11(d) shows the clustering results obtained by the SVC algorithm with q = 0.001 and C =0.01. The result obtained by



(a) *(b)*

Figure 13: (a) The clustering results of DBSCAN. (b) The clustering results of OPTICS.

the HRE method is shown in Fig. 12(a). There were 1201 data points retained in the dataset. The execution time of the HRE method was 58.13 sec. By setting q = 0.001 and C = 0.01, we obtained the final cluster result that is shown in Fig. 12(b). Figure 13 illustrates the clustering results of DBSCAN and OPTICS. Our proposed method can find the correct cluster number but HRE, DBSCAN and OPTICS cannot do so for different parameter selections.

The simulation results of the benchmark datasets confirm that our proposed methods can correctly identify the cluster numbers as well as the cluster boundaries that are not altered by the data preprocessing procedure. However, the cluster results of the benchmark datasets produced by HRE, DBSCAN and OPTICS show that these methods are sensitive to cluster densities and the amount of noise contained in the dataset.

3) Wisconsin Breast Cancer Dataset

The Wisconsin breast cancer dataset [Black et al. 1998] contains 699 cases of diagnostic samples, and each sample contains nine features. After the removal of the 16 samples with missing values, there are a total of 683 data patterns belonging to benign (444 samples) and malignant tumors (239 samples). We set k_1 = 55 and k_2 = 10 in this example and obtained α = 20.8501, δ = 7.7513, and λ_p = 3.25 from (10), (11), and (15), respectively. We used the SNN algorithm to eliminate the noise points. In this step, a total of 117 data points were eliminated. Next, we used the concept of unit vectors to eliminate the core points and 203 data points were retained. The execution time of our proposed method was 5.37 sec. Finally, we set q = 0.03 and C = 0.01 to obtain the final clustering results using the SVC algorithm. The classification accuracy of the Wisconsin breast cancer dataset was 96.57% by our proposed method. The result of the data preprocessing step using the HRE method, 203 data points were retained in the dataset. The execution time of the HRE method was 7.06 sec. By setting q = 0.03 and C = 0.01, the classification accuracy of the Wisconsin breast cancer dataset was 93.25%. The performance of our proposed method was better than the HRE method in this example. From the result of this example, we believed that our proposed approach can be served as an effective tool in dealing with classification problems.

5 Conclusions

This paper presents an efficient data preprocessing procedure that ameliorates the limitations of SVC for large datasets. Our approach can eliminate insignificant data points from the training datasets without significantly altering the final cluster configuration. The preprocessing procedure utilizes a shared nearest neighbor (SNN) algorithm for eliminating the noise points, and the concept of unit vectors for removing the core points from the datasets. Our simulation results have successfully validated the effectiveness of the proposed method for improving the capability of SVC in dealing with large datasets. Our future research includes the verification of our proposed method on different real-world applications.

References

[Ankerst et al. 1999]Ankerst, M., Breunig, M., Kriegel, H. P., Sander, J.: “OPTICS: Odering Points to Identify the Clustering Structure”; In Proc. ACM SIGMOD Int. Conf. on Management of Data (1999), 49-60.

[Ben-Hur et al. 2000] Ben-Hur, A., Horn, D., Siegelmann, H. T., Vapnik, V.: “A Support Vector Clustering Method”; In Proc. of Int. Conf. on Pattern Recognition (2000), 724-727.

[Black et al. 1998] Blake, C., Keogh, E., Merz, C.: “UCI Repository of Machine Learning Databases”. http://www.ics.uci.edu/~mlearn/MLRepository.html. University of California, Irvine, Department of Information and Computer Sciences (1998).

[Cortes and Vapnik 1995]Cortes, C., Vapnik, V.: “Support-Vector Network,” Machine Learning, 20 (1995), 273-297.

[Daszykowski et al. 2001]Daszykowski, M., Walczak, B., Massart, D. L.: “Looking for Natural Patterns in Data Part 1. Density-Based Approach”; Chemom. Intell. Lab. Syst., 56 (2001), 83-92.

[Daszykowski et al. 2002] Daszykowski, M., Walczak, B., Massart, D. L.: “Looking for Natural Patterns in Analytical Data 2. Tracking Local Density with OPTICS”; J. Chem. Inf. Comput. Sci., 42 (2002), 500-507.

[Ertöz et al. 2003] Ertöz, L., Steinbach, M., Kumar, V.: “Finding Clusters of Different Sizes, Shapes, and Densities in Noisy, High Dimensional Data”; In Proc. of SIAM Int. Conf. on Data Mining (2003), 1-12.

[Ester et al. 1996]Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”; In Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (1996), 226-231.

[Guha et al. 1998]Guha, S., Rastogi, R., Shim, K.: “CURE: An Efficient Clustering Algorithms for Large Databases”; In Proc. ACM SIGMOD Int. Conf. on Management of Data (1998), 73-84.

[Guha et al. 1999]Guha, S., Rastogi, R., Shim, K.: “ROCK: A Robust Clustering Algorithms for Categorical Attributes”; In Proc. of IEEE Conf. on Data Engineering (1999), 512-521.

[Jarvis and Patrick 1973] Jarvis, R. A., Patrick, E. A.: “Clustering Using a Similarity Measure Based on Shared Nearest Neighbors”; IEEE Trans. Computers, C-22, 11 (1973), 1025-1034.

[Karypis et al. 1999] Karypis, G., Han, E. H., Kumar, V.: “Chameleon: A Hierarchical Clustering Algorithm Using Dynamic Modeling”; Computer, 32, 8 (1999), 68-75.

[Lee and Lee 2005] Lee, J., Lee, D.: “An Improved Cluster Labeling Method for Support Vector Clustering,” IEEE Trans. Pattern Analysis and Machine Intelligence, 27 (2005), 461-464.

[Saketha Nath and Shevade 2006] Saketha Nath, J., Shevade, S. K.: “An Efficient Clustering Scheme Using Support Vector Methods”; Pattern Recognition, 39, 8 (2006), 1473-1480.

[Wang and Chiang 2008a]Wang, J.-S., Chiang, J.-C.: “A Cluster Validity Measure with Outlier Detection for Support Vector Clustering”; IEEE Trans. Systems, Man, and Cybernetics-Part B, 38, 1 (2008), 78-89.

[Wang and Chiang 2008b]Wang, J.-S., Chiang, J.-C.: “A Cluster Validity Measure with a Hybrid Parameter Search Method for Support Vector Clustering Algorithm”; Pattern Recognition, 41, 2 (2008), 506-520.

[Yang et al. 2002] Yang, J., Castro, V. E., Chalup, S. K.: “Support Vector Clustering Through Proximity Graph Modeling”; In Proc. of 9th Int. Conf. on Neural Information Processing (2002), 898-903.

[Zhang et al. 1996]Zhang, T., Ramakrishnan, R., Linvy, M.: “BIRCH: An Efficient Data Clustering Method for Very Large Databases”; In Proc. ACM SIGMOD Int. Conf. on Management of Data (1996), 103-114.