Validation-Driven Algorithm for SVC

Keng-Hsuan Wu
Department of Electrical Engineering
National Taiwan Ocean University
Keelung, Taiwan
woods@softcomp.ee.ntou.edu.tw

Jung-Hua Wang†
Department of Electrical Engineering
National Taiwan Ocean University
Keelung, Taiwan
jhwang@mail.ntou.edu.tw

Abstract—This paper presents a novel approach, called validation-driven algorithm (VDA) for improving support vector clustering (SVC). The SVC algorithm is a well-known kernel-based clustering approach, but its clustering results keenly rely on the proper choice of parameters of kernel functions. Particularly, we have found that performance of SVC is very sensitive to the initial value of the kernel parameter \( q_{ini} \), and a poor initial \( q \) value inevitably will incur heavy computation time for obtaining a satisfactory result. Furthermore, it is very difficult to determine the proper \( q \) value for input containing clusters with large diversity in variations. Previous SVC algorithms use an identical trial-and-error \( q \) for all clusters. Aiming to tackle these issues, VDA not only capable of finding a good initial value of the \( q \) parameter, but also provides a computationally efficient procedure for verifying the validity of the parameter setting as well as the clustering result. Simulation results demonstrate the effectiveness of the proposed VDA algorithm.

I. INTRODUCTION

Recently, Support Vector Machines (SVMs), motivated by statistical learning theory, have drawn great attention in the field of machine learning [1][2][11]. Unlike other well-known networks, such as multilayer perceptron (MLP) and radial-basis function (RBF) networks, training an SVM is equivalent to solving a linearly constrained convex quadratic programming problem. Since the MLP and RBF networks rely on the minimization of a nonlinear error function which may be nonconvex, the local minima problem in training can be avoided by using the SVM approach. The approach of Support Vector Clustering (SVC) is originated from the concept of SVMs which use support vectors to form several contours. In its most basic form, a SVC algorithm consists of two main stages. In the first stage, input raw data are mapped into feature space by a nonlinear kernel function for identifying the smallest radius of the sphere that encloses all the data points by support vectors. Among all kinds of kernel functions, Gaussian function is the most frequently used. In the second stage, the sphere so obtained is transferred back to the input data space, which forms several contours corresponding to cluster boundaries.

In fact, these two stages say for themselves why SVC algorithms are advantageous over other clustering algorithms. Namely, they are capable of generating cluster boundaries of arbitrary shaped and able to deal with outliers by employing a soft margin constant that allows the sphere in feature space not to enclose all points.

Despite all these merits, there are some issues regarding the kernel parameter of SVC that remain to be solved. One of them is how to choose a suitable kernel parameter for the adopted kernel function. Take the Gaussian function as an example, changing the value of the kernel width \( q \) will greatly affect the number of resulting clusters. That is, the larger the value of \( q \) is, the more the number of clusters will be obtained. In some clustering methods where SVM is used as a “divisive” algorithm that starts with a small \( q \) and gradually increases it [3][4]. For example, it might start the value of \( q \) with 0.5, even if the suitable value of \( q \) of the SVC algorithm is actually 10. Apparently such process of finding a suitable value of \( q \) is rather slow. Although Ben-Hur et al. [3] have proposed a rule using the concept of data scale to determine the initial value of \( q \), this rule may not be efficient and applicable for most real data. Furthermore, in cases where input clusters have large diversity in variations, it is desirable to have a way of determining the suitable values of \( q \) for each local cluster terrain so that accurate labeling can be ensured.

In light of the foregoing observations, this paper presents a validation-driven algorithm, VDA, for improving the performance of SVC. Key attributes of VDA is that by setting an appropriate initial value, one can achieve satisfactory clustering result more quickly, and moreover, thorough the use of a validation checking procedure, we not only can avoid the erroneous labeling, but also can deal with input data containing diverse variations among clusters. The rest of this paper is organized as follows. Some background necessary to understand the work is given in Section II. In Section III, we elaborate the proposed method of VDA, focusing on setting the appropriate initial \( q \) value and applying the proposed validation checking procedure to determine the suitable \( q \) value for each separate cluster in the input data. Simulation results are given in Section IV for
verifying the effectiveness of VDA. Finally, conclusions are given in Section V.

II. THE SVC ALGORITHM

Mathematical formulation of SVC is briefly stated as follows. Let \( \{x_i\} \subseteq \chi \) be a data set of \( N \) points, with \( \chi \subseteq \mathbb{R}^d \), the data space. A nonlinear mapping function \( \Phi \) is used to map \( \chi \) into a high-dimensional feature space. The objective is to search for the smallest enclosing sphere of radius \( R \) expressed in the following optimization problem:

\[
\min R^2 + C \sum_j \xi_j
\]

subject to \( \| \Phi(x_j) - a \|^2 \leq R^2 + \xi_j \quad \forall j, \quad (1) \)

where \( a \) is the center and \( \xi_j \) are slack variables allowing for soft boundaries. That is, they allow some data points lying out of the sphere. The above problem is usually solved by the Lagrangian function using its dual problem:

\[
L = R^2 - \sum_j (R^2 + \xi_j - \| \Phi(x_j) - a \|^2) \alpha_j - \sum_j \xi_j \beta_j + C \sum_j \xi_j
\]

where \( \alpha_i \geq 0 \) and \( \beta_i \geq 0 \) are the Lagrangian multipliers, \( C \) is a constant. The Karush-Kuhn-Tucker (KKT) conditions allow the problem to be rewritten as

\[
\max W = \sum_j \Phi(x_j)^2 \alpha_j - \sum_{ij} \alpha_i \alpha_j \Phi(x_i) \cdot \Phi(x_j)
\]

subject to \( 0 \leq \alpha_j \leq C, \sum_j \alpha_j = 1, \quad j = 1, \ldots, N, \quad (3) \)

where the dot product \( \Phi(x_i) \cdot \Phi(x_j) \) represent an appropriate Mercer kernel \( K(x_i, x_j) \). In literature the most widely seen kernel function is the Gaussian type

\[
K(x_i, x_j) = e^{-\|x_i - x_j\|^2/q}
\]

with width parameter \( q \). As noted in [3], polynomial kernels do not yield tight contours representations of a cluster. The Lagrangian \( W \) is now written as

\[
\max W = \sum_j K(x_j, x_j) \alpha_j - \sum_{ij} \alpha_i \alpha_j K(x_i, x_j)
\]

subject to \( 0 \leq \alpha_j \leq C, \sum_j \alpha_j = 1, \quad j = 1, \ldots, N, \quad (5) \)

We can optimize Eq.(4) to obtain the Lagrangian multipliers \( \alpha_i \) using Sequential Minimal Optimization (SMO) algorithm or other learning algorithms [5][6][12]. There exist data points lying on the surface of the sphere in feature space, and these points are referred to as support vectors (SVs). Here,

\[
0 < \alpha_i < C \quad \forall i \in \text{SVs}
\]

And when \( \alpha_i = C \), the data points lie outside the feature space, called bounded support vectors (BSVs). At each point \( x \) we define the distance of its image in feature space from the center of the sphere:

\[
R^2(x) = ||\Phi(x) - a||^2
\]

Also, according to KKT condition, we can acquire Eq.(7) from Eq.(2)

\[
a = \sum_j \alpha_j \Phi(x_j)
\]

In view of Eq.(7) and the definition of the kernel, we have

\[
R^2(x) = K(x,x) - 2 \sum_j \alpha_j K(x_j,x) + \sum_{ij} \alpha_i \alpha_j K(x_i,x_j)
\]

The radius of the sphere is

\[
R=\{\Phi(x) | x \text{ is a support vector}\}
\]

In practice, its average over all support vectors is used as the radius \( R \). The cluster boundaries are then formed by connecting the data points that satisfy \( R(x) = R \). The SVs, BSVs, and all other points are located on the boundaries, the outside of the boundaries, and the inside of the boundaries, respectively. The cluster description itself does not differentiate points that belong to different clusters. In order to label data points into clusters, an adjacency matrix \( A_q \) is defined to identify the components of a cluster. Define the adjacency matrix \( A_q \) between pairs of points \( x_i \) and \( x_j \), whose images lie in or on the sphere in feature space:

\[
A_q = \begin{cases} 
1 & \text{if for all } y \text{ on the line segment connecting } x_i \text{ and } x_j, \ R(y) \leq R \\
0 & \text{otherwise} 
\end{cases}
\]

Clusters are now defined as the connected components of the graph induced by \( A \). And the values of \( y \) can be obtained by sampling a number of points from the line segment connecting \( x_i \) and \( x_j \). Note that BSVs are unclassified by this procedure since their feature space lie outside the enclosing sphere.

More importantly, the key issue of SVC is about the kernel parameter. Because the number of clusters is governed by the value of \( q \), the selection of the value \( q \) will be the first priority. Also, since Ben-Hur proposed the SVC algorithm as the “divisive” clustering algorithm, which means starting from a small value of \( q \) and gradually increasing it, properly setting the initial value of \( q \) naturally becomes a key issue. Although Ben-Hur indicated the initial value of \( q \) can be chosen as

\[
q_{ini} = \frac{1}{\max_{\|y\|} \|x_i - y\|^2}
\]

it cannot properly deal with many general input data. For example, Fig.1 shows the SVC result using \( q=5 \) (initially \( q_{ini}=1 \)), the test input used is three Gaussian clusters with scale ranging from -0.5 to 0.5. To compare, in Fig.2 the three
clusters still have the same scale range but with larger variance of 0.2 than in Fig.1. If one applied Eq.(10) to this input, \( q_{ini} \) will be 1, and then increased to \( q=50 \). We found that when the input variance is large as in Fig.2, the number of clusters would be erroneously less than 3 if \( q<50 \). This implies that heavy computation load is inevitable if the input contains clusters with large variance and \( q_{ini} \) is small. Consequently, it is essential to have a good initial \( q \) value in order to save computation time.

For more complex data as shown in Fig. 3 where we see three Gaussian distributions of varying sizes and different number of data points in each cluster, SVC cannot perform well. Because increasing the value of \( q \) results in more number of clusters, and even worse, it cannot guarantee that the clusters generated by setting larger value of \( q \) are correct. Specifically, if one desires to obtain correct labeling for the two Gaussians with smaller variances in Fig.3, the value of \( q \) has to be larger. Unfortunately, using a large \( q \) value will generate more spurious clusters, as shown in Fig.4 where \( q=600 \) is used. Conversely, Fig.5 depicts the inaccurate result from using a smaller value of \( q \) for the two Gaussian data.
III. VALIDATION-DRIVEN ALGORITHM

To solve the aforementioned problems of setting an appropriate $q_{ini}$ and determining suitable $q$ values for each cluster separately, we propose the VDA method and the flowchart of which is shown in Fig.6. VDA mainly consists of the following processes: setting an initial value of $q$, checking validity of homogeneity and separation, and finally the decision rule.

A. Setting $q_{ini}$

In [3], it has been shown that the initial value of $q$ depends on the scale of data. But from Fig.1 and Fig.2, we can see that the value of $q$ is not only related to data scale, but the variance of data. The larger variance the data has, the larger the value of $q$. Thus, an essential step in VDA is to sample the input data and estimate the variance of data. For the former, one can use Vector Quantization [7] techniques, such as LBG algorithm [8] or Self-Organization Map (SOM) [9]. Following that, we apply SVC to the sample data. The purpose of doing this is to find rough clusters, which can be used to determine the variances of these clusters. Hence, the value of $q$ can be chosen by using Eq.(10). Finally,

$$q_{ini} = \frac{1}{k} \sum_{m=1}^{k} \frac{1}{\max_{j} \| x_{jm} - \bar{x}_{jm} \|^{2}} \times (\sigma_{m}^{2} \times factor)$$

(11)

$$factor = \begin{cases} 1 & \text{if } \sigma_{m}^{2} \geq 1 \\ a \times 10^{b} & \text{if } \sigma_{m}^{2} = a \times 10^{-b}, a > 0 \text{ and } b > 0 \end{cases}$$

(12)

where $k$ denotes the number of clusters of the sample data, $\sigma_{m}^{2}$ the variance of the $m$ cluster and the factor is to prevent the variance from being less than 1.

B. Validity measure

Validity measure is used to guide the SVC algorithm to determine a suitable parameter of the kernel functions [4], which can be measured in terms of homogeneity and separation.

First, we apply Minimum Spanning Tree (MST) algorithm [10] to every data points of each cluster. Given an undirected graph $G = (V, E)$, where $V$ denotes the set of vertices and $E$ the set of edges and a real number $w(e)$ for each edge $e \in E$ called the weight of edge $e$, the MST is the spanning tree $T^{*}$ on $G$ such that $w(T^{*}) = \sum_{e \in T^{*}} w(e)$ is the minimum taken over all possible spanning trees of $G$. The feasibility of applying MST is based on the fact that by setting every input vector as a vertex and connecting them in pursuit of minimal total weights (distances). And the goal of applying MST here is to find the connection length between data points. Intuitively, the higher intra-class homogeneity is, the smaller $w(T^{*})$ will be obtained. In the following, we define the homogeneity and separation validity measures.

To define the homogeneity of each cluster, we first determine and find the mean weight $m_i$, the variance $\sigma_i^2$, and the largest weight $L_i$ of each cluster $i$ after applying MST,

$$m_i = \frac{1}{n} \sum_{e \in T^{*}} w_i(e)$$

(13)

$$\sigma_i^2 = \frac{1}{n-1} \sum_{j=1}^{p} (w_{ij}(e) - m_i)^2$$

(14)

$$L_i = \max_{j} w_{ij}(e) \quad \forall j$$

(15)

where $p$ denotes the number of weights in each cluster. Then we check if $L_i > m_i + 2\sigma_i$ or $L_i < m_i - 2\sigma_i$, if so, the homogeneity of the cluster in question is doubtful. The target cluster will be marked as $H_i$ for the subsequent decision rule.

To define the separation of inter-cluster, the intercluster distance is denoted as $f_{ij}(C_i,C_j)$, it is the distance between the SVs in cluster $C_i$ and those in cluster $C_j$.

$$f_{ij}(C_i,C_j) = \min \{ f(x_i,x_j) ; x_i \text{ are SVs } \in C_i, x_j \text{ are SVs } \in C_j \}$$

(16)

We then check if $f_{ij}(C_i,C_j) < m_i$ and $f_{ij}(C_i,C_j) < m_j$, if so, the separation between cluster $i$ and cluster $j$ is in doubt. So we mark the two clusters as $S_i$ for the subsequent decision rule. Note that cluster containing only one point will not be taken into account in this procedure and marked as $O_i$.

After marking the clusters, we have to make decision for these clusters. Firstly, clusters that do not comply with the validity measure will reset its value of $q$ and re-run the SVC algorithm, one by one. Particularly, cluster marked as $H_i$ will use a larger value of $q$, and clusters $S_i$ will use a smaller value of $q$. The increment or decrement in $q$ relies on the initial value $q_{ini}$. For example, if $q_{ini}$ is 0.1, the degree of increment or decrement is $10^3$. Because we will not consider the outlier, any $O_i$ will be merged with the nearest cluster. After applying the SVC algorithm to the marked clusters, we will obtain a new clustering result. The resulting clusters along with the previously unmarked clusters will simultaneously participate in the validation of homogeneity and separation. The iterative process will be terminated until no more marked clusters.

IV. SIMULATION RESULTS

In this section simulation results are demonstrated to show the performance of the proposed VDA method. We tested the input data in Fig.3 containing three Gaussian distributions of varying sizes and different number of data points in each cluster. Firstly, the sampling data were picked using uniformly random method. Then the initial value of $q$ was determined to be 110 by using Eq.(11). Fig.7 depicts the first iteration applying $q=110$. Through the validation checking procedure, the two Gaussian data with smaller variances will be marked. In the next iteration, we apply a
larger value of $q$ to SVC for the marked clusters. Table I lists the clustering results with different $q$ values, where $q_{ij}$ represents the $j$ marked cluster at the $i^{th}$ iteration. Moreover, result in Fig. 8 has verified that the VDA is indeed effective.

We also applied different data type for examining the effectiveness of VDA, as shown in Fig. 9, Fig. 10 and Fig. 11.

![Fig. 6. The flowchart of VDA](image)

![Fig. 7 The result of the first iteration with VDA, $q_{ini}=110$](image)

![Fig. 8 The result of the final iteration with VDA, $q_{ini}=510$](image)

![Fig. 9 The result of the various data with VDA, $q=150$](image)

<table>
<thead>
<tr>
<th>$q$</th>
<th>Cluster number</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_{ini}=110$</td>
<td>2</td>
</tr>
<tr>
<td>$q_{i1}=210$</td>
<td>2</td>
</tr>
<tr>
<td>$q_{i2}=310$</td>
<td>2</td>
</tr>
<tr>
<td>$q_{i3}=410$</td>
<td>2</td>
</tr>
<tr>
<td>$q_{i4}=510$</td>
<td>3</td>
</tr>
</tbody>
</table>

**TABLE I.**

**COMPARISON OF USING DIFFERENT $q$'s**
V. CONCLUSIONS

We have presented the Validation-Driven Algorithm (VDA) which is effective in improving Support Vector Clustering. The proposed VDA incorporates the validation checking procedure to measure whether a separate cluster complies with the homogeneity, and at the same time whether two neighboring clusters comply with the separation measure. According to the decision rule, we can pick out the marked (doubtful) clusters and apply them to the SVC algorithm with a new value of $q$. The simulation results have demonstrated the effectiveness of proposed VDA approach, and its capability of automatically identifying the suitable value of $q$ for each separate valid cluster. In the future, we consider to apply other validity measures [14][15][16]. Also, as another key step in the SVC algorithm is the cluster assignment of each data point. Different labeling techniques could generate different results. Hence, we will also focus on trying other labeling techniques [13].

VI. ACKNOWLEDGEMENT

This research was supported by the National Science Council of Taiwan under grant: NSC 93-2611-E-019-007.

VII. REFERENCES