An efficient algorithm to improve the accuracy and reduce the computations of LS-SVM

M. Baymani* and A. Mansoori

Abstract

We present a novel algorithm, which is called Cutting Algorithm (CA), for improving the accuracy and reducing the computations of the Least Squares Support Vector Machines (LS-SVMs). The method is based on dividing the original problem to some subproblems. Since a master problem is converted to some small problems, this algorithm has fewer computations. Although, in some cases that the typical LS-SVM cannot classify the dataset linearly, applying the CA the datasets can be classified. In fact, the CA improves the accuracy and reduces the computations. The reported and comparative results on some known datasets and synthetics data demonstrate the efficiency and the performance of CA.


Keywords: Least squares support vector machine; Cutting algorithm; Classification.

1 Introduction

Support Vector Machines (SVMs) were introduced by Vapnik in 1995 [14, 15] within the area of statistical learning theory. SVMs are very popular and powerful in learning systems. Over the years, a variety of numerical optimization algorithms for SVM learning have been proposed [5, 12, 7]. However, these traditional algorithms may not be applicable for digital computers since

*Corresponding author
Received 31 August 2018; revised 15 August 2019; accepted 15 October 2019
Mojtaba Baymani
Department of Computer and Mathematics, Quchan University of Advanced Technology, Quchan, Iran. e-mail: m.baymani@qut.ac.ir
Amin Mansoori
Department of Applied Mathematics, Ferdowsi University of Mashhad, Mashhad, Iran. e-mail: a-mansoori@um.ac.ir
the computing time required for a solution is greatly dependent on the dimension and the structure of the problem and the complexity of the algorithm used. LS-SVMs are least-squares versions of SVMs, which are a set of related supervised learning methods that analyze data and recognize patterns, and which are used for classification and regression analysis. In this version, one finds the solution by solving a set of linear equations instead of a convex quadratic programming (QP) problem for classical SVMs. Least-squares SVM classifiers were proposed by Suykens and Vandewalle [13]. The LS-SVM is modified from SVM, which can be used to approximate the nonlinear system with higher accuracy [13, 3, 8, 16, 10, 4]. With better performance than SVM, LS-SVM model has been successfully applied in diverse fields, such as CO concentration, income, precipitation, wind speed, and so on. In the original space, the LS-SVM with equality constraints can be expressed as follows:

\[
\min \frac{1}{2}||w||^2 + \frac{C}{2} \sum_{i=1}^{l} \eta_i^2 \\
\text{s.t.} \quad y_i = w \cdot \varphi(x_i) + b + \eta_i, \quad i = 1, 2, \ldots, l, \tag{1}
\]

where \( S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_l, y_l)\} \) is a set of \( l \) training samples, \( x_i \in \mathbb{R}^m \) is an \( m \)-dimensional sample in the input space, \( y_i \in \{-1, +1\} \) is the class label of \( x_i \), \( w \in \mathbb{R}^m \) and \( b \) are weight vector and bias, respectively, and \( C \) is a positive and sufficiently large parameter and indicates the regularization parameter. Also, \( \eta_i \) indicates the slack variable. Inputs to the SVM system are the training data and the constant. The system will calculate proper slack variables \( \eta_i \) and will determine the separating hyperplane. Moreover, \( \eta_i \) is the training error corresponding to data sample \( x_i \). Also, the quantity \( C \eta_i^2 \) is the “penalty” for any data point \( x_i \) that either lies within the margin on the correct side of the hyperplane \( \eta_i \leq 1 \) or on the wrong side of the hyperplane \( \eta_i > 1 \). Increasing the values of slack variables, helps in reducing the effect of noisy support vectors. SVMs find the optimal separating hyperplane with the minimal classification errors. Let \( w \) and \( b \) denote the optimum values of the weight vector and bias, respectively. The hyperplane can be represented as: \( w^T x + b = 0 \), that \( w = [w_1, w_2, \ldots, w_m] \) and \( x_i = [x_{i1}, x_{i2}, \ldots, x_{im}] \); \( w \) is the normal vector of the hyperplane, and \( b \) is the bias.

Using the nonlinear function \( \varphi \), the data are mapped from the input feature space to a higher-dimensional space. The Lagrange function of (1) similar to [1] can be built by

\[
L(w, b, \eta, \lambda) = \frac{1}{2}||w||^2 + \frac{C}{2} \sum_{i=1}^{l} \eta_i^2 + \sum_{i=1}^{l} \lambda_i (y_i - w \cdot \varphi(x_i) - b - \eta_i), \tag{2}
\]

where \( \lambda_i \) denotes the Lagrange multiplier. The optimal point will be in the saddle point of the Lagrangian function, and then we obtain
\[
\begin{align*}
\frac{\partial L}{\partial w} &= 0 \implies w = \sum_{i=1}^{l} \lambda_i \varphi(x_i), \\
\frac{\partial L}{\partial b} &= 0 \implies w = \sum_{i=1}^{l} \lambda_i = 0, \\
\frac{\partial L}{\partial \lambda_i} &= 0 \implies \lambda_i = -C \eta_i, \\
\frac{\partial L}{\partial \eta_i} &= 0 \implies y_i - w \varphi(x_i) - b - \eta_i = 0.
\end{align*}
\] (3)

Thus, the optimization problem (1) can be transformed into the following linear system by eliminating the vectors \( \omega \) and \( \eta \):

\[
\begin{bmatrix}
0 \\
Q \Lambda + \frac{1}{C} I
\end{bmatrix}
\begin{bmatrix}
\eta \\
\Lambda
\end{bmatrix}
= \begin{bmatrix}
0 \\
Y
\end{bmatrix},
\]

where \( Q = [1, \ldots, 1]^T \), \( \Lambda = [\lambda_1, \ldots, \lambda_l]^T \), and \( Y = [y_1, \ldots, y_l]^T \). Also, the kernel function can be set as:

\[
K(x_i, x_j) = \phi^T(x_i)\phi(x_j).
\]

Two parameters are required in the LS-SVM model selection, which are the bandwidth of the Gauss radial basis kernel \( \sigma \) and the regularization parameter \( C \). In SVMs, the computational complexity always is a big problem in training stage for sparse data. These complexities reduce accuracy in SVMs. This problem is greater in LS-SVM; what that it is not sparse. Therefore, we should solve a system of linear equations to eliminate this problem. In this way, if the number of training data’s increases, then the computational complexity of the system of linear equations improves. However, by using the (Cutting Algorithm) CA, we try to improve the efficiency and reduce the computations of the LS-SVM.

Motivated by the former discussion, in this text, we propose a novel algorithm for solving this problem; we call this algorithm as CA. The CA reduces computations in training stage for variety of SVMs and also improves the accuracy. We use the CA besides LS-SVMs on training set of samples. Our idea in this algorithm is to break main problem to smaller problems and solve each of them separately. As we know, the LS-SVM cannot classify the nonlinear datasets linearly; however, we use the proposed algorithm to classify the nonlinear datasets, linearly as well. In addition, we compare the proposed method with some other known methods.

The paper is organized as follows. Next section, the viewpoint of the CA is stated. We describe the proposed CA in this section. Section 3 explains the geometry illustration of the CA with one cut. The CA in general case is studied in Section 4. Section 5 investigates the CA in \( n \)-stage cut. In this section, we discuss some algorithms for the \( n \)-stage CA. The computational results are given in Section 6. Also, comparative results are obtained here. Finally, Section 7 states the conclusions.
2 Viewpoint of cutting algorithm (CA)

In this section, we describe the CA. In fact, we find a hyperplane dividing the training set into two subsets. For finding this hyperplane, we call each sides of the hyperplane a dimension. Consequently, we choose two dimensions \( r \) and \( s \). On any of these dimensions, by getting the average of vectors of any class, we may obtain (the average points are denoted by \( x_{rs+} \) and \( x_{rs-} \)) the \( x_{rs+} \) and the \( x_{rs-} \) in the \( rs \) hyperplane. It is obvious that, the point \( x_{rs+} \) is in the positive and the point \( x_{rs-} \) is in the negative class of vectors on the \( rs \) hyperplane. The passing line from \( x_{rs+} \) and \( x_{rs-} \) decomposes the positive class and the negative class into two disjoint parts. Therefore, it exists a hyperplane passing from the given two average points on \( \mathbb{R}^n \) and it is perpendicular on \( rs \) hyperplane. The equation of this hyperplane that its normal vector has two nonzero terms, can be considered as \( w_r x_r + w_s x_s + b_m = 0 \) in \( \mathbb{R}^n \). This hyperplane divides the set \( S \) (the training set) into two training sets \( S_u \) and \( S_d \) as follows:

\[
S_u = \{(x_i, y_i) \in S \mid w_r x_{ir} + w_s x_{is} + b_m \geq 0\},
\]
\[
S_d = \{(x_i, y_i) \in S \mid w_r x_{ir} + w_s x_{is} + b_m < 0\}.
\]

As stated before, we can obtain \( S_u \) and \( S_d \) from \( S \). As a matter of fact, we have the following subproblems:

\[
\begin{aligned}
\min & \quad \frac{1}{2} \|\omega\|^2 + \frac{C}{2} \sum_{i=1}^{l_u} \eta_i^2 \\
\text{s.t.} & \quad y_i - \omega \cdot \varphi(x_i) - b = \eta_i, \quad (x_i, y_i) \in S_u, \quad i = 1, 2, \ldots, l_u, 
\end{aligned}
\]

and

\[
\begin{aligned}
\min & \quad \frac{1}{2} \|\omega\|^2 + \frac{C}{2} \sum_{i=1}^{l_d} \eta_i^2 \\
\text{s.t.} & \quad y_i - \omega \cdot \varphi(x_i) - b = \eta_i, \quad (x_i, y_i) \in S_d, \quad i = 1, 2, \ldots, l_d,
\end{aligned}
\]

where \( l_u \) and \( l_d \) are numbers corresponding to \( S_u \) and \( S_d \), respectively. Also, it is clear that \( l_u + l_d = l \). In this approach, since the average is done on two dimensions, so by a little computation, we can find the desirable disjoin hyperplane, where its normal vector has two nonzero terms. Moreover, simplicity of this algorithm, simplifies the work of separating the training sets. The hyperplane dividing the training vector's set is called the cutting hyperplane.
3 Geometry illustration of CA with one stage cut

For the first step, we consider one cut. We are going to disjoint the problem into two parts. Also, we denote the training vector’s set in positive with \( X_p \) and the negative class with \( X_n \), respectively. As a matter of fact, we have

\[
X_p = \{ x_i | (x_i, y_i) \in S, y_i = +1 \}, \tag{6}
\]
\[
X_n = \{ x_i | (x_i, y_i) \in S, y_i = -1 \}. \tag{7}
\]

By the former discussion, the following algorithm (Algorithm 1) is given as follows:

\begin{algorithm}
1. Input the training set \( S \) with \( l = 2 \).
2. Obtaining the average \( X_p \) and \( X_n \) and denote them by vectors \( x_{r++} \) and \( x_{r--} \), respectively.
3. Obtaining the hyperplane \( w_m \cdot x + b_m = 0 \) that passes from the given two average points \( x_{r++} \) and \( x_{r--} \).
4. Determine two training sets \( S_u \) and \( S_d \) for one cut as follows:
   \[
   S_u = \{ (x_i, y_i) \in S \mid w_m x_i + b_m \geq 0 \}, \\
   S_d = \{ (x_i, y_i) \in S \mid w_m x_i + b_m < 0 \}.
   \]
5. Using an SVM to determine decision functions \( f_u(x) = \text{sign}(g_u(x)) \) and \( f_d(x) = \text{sign}(g_d(x)) \) for \( S_u \) and \( S_d \), respectively.
6. Output the decision function as follows:
   \[
   f(x) = \begin{cases} 
   f_u(x), & w_m x_i + b_m \geq 0, \\
   f_d(x), & w_m x_i + b_m < 0.
   \end{cases}
   \]
\end{algorithm}

For demonstrating the accuracy and the performance, we test CA to classify some training set for some given (synthetics) data. Figures 1–6 show the results. In all results, we consider the following hypotheses:

1. All of positive and negative classes have 100 elements.
2. Symbols + and * refer to the positive and the negative classes, respectively.
3. Line (0) is the separable line \( w.\varphi(x) + b = 0 \).
4. Line (−1) is the separable line \( w.\varphi(x) + b = -1 \).
5. Line (+1) is the separable line \( w.\varphi(x) + b = +1 \).
6. For all problems, we set $C = 10$.

7. We used the Radius Basis Function (RBF) kernel with $\sigma = 1$ in nonlinear problems (i.e., $K(x, x_i) = e^{-\frac{||x-x_i||^2}{\sigma^2}}$).

Figure 1 shows the classification of some data with LS-SVM. In Figure 2 for the same problem, we use the CA in LS-SVM to classify the data. One can check that the classification is done accurately.

**Figure 1:** Geometric interpretation of LS-SVM with imbalance data

**Figure 2:** Geometric interpretation of LS-SVM in dominant problem by using CA

Moreover, Figures 3–6 demonstrate the interpretation in order to classify for nonlinear problems. Figures 3 and 4 show the interpretation of a problem to classify with LS-SVM and LS-SVM with CA, respectively. Also, Figures 5 and 6 show the interpretation of classification of another dominate problem.
with LS-SVM and LS-SVM with CA, respectively. In none of the test problems the LS-SVM cannot classify the data accurately, however, the LS-SVM with CA does that as well.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3}
\caption{Geometric interpretation of nonlinear LS-SVM in dominant problem}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure4}
\caption{Geometric interpretation of nonlinear LS-SVM in dominant problem by using CA}
\end{figure}

4 CA with one cut in general status

Here, we study the CA with one cut in general case. In this case, in high dimensions, again we get the average just on two dimensions. Consequently, this work does not increase the computations. Therefore, the efficiency of this algorithm improves by increasing the training vector’s dimensions. The
the training vectors distributed uniformly on that dimension.

5 CA with n-stage cuts

In this section, the CA with n-stage cut is investigated. After one cut on training series \( T \), it separates into two subsets \( T_0 \) and \( T_1 \). It is clear that, the subsets \( T_0 \) and \( T_1 \) can be considered as new sets, so they can be cut again. We denote the sets obtained from cutting \( T_0 \) by \( T_{00} \) and \( T_{01} \). Also, the sets obtained from cutting \( T_1 \) is denoted by \( T_{10} \) and \( T_{11} \). These four disjoint sets are on third stage. Do this procedure for \( n \) stage, we have the following sets:

\[
T_{i_1, i_2, \ldots, i_m}, \quad m = 1, 2, \ldots, n - 1, \quad i_1, i_2, \ldots, i_m \in \{0, 1\}.
\]  

(8)
Algorithm 2 The cutting algorithm in two dimensions with one stage cut in general case
1: Input the training series $S$ with an arbitrary $t \geq 2$.
2: Choose two dimensions $r$ and $s$ of $\mathbb{R}^n$.
3: Obtaining the average $X_p$ and $X_m$ in $rs$ hyperplane and denote them by vectors $x_{r+s}$ and $x_{r-s}$, respectively.
4: Obtaining the hyperplane $w_r x_r + w_s x_s + b_m = 0$ where passes from the given two average points $x_{r+s}$ and $x_{r-s}$.
5: Determine two training series $S_u$ and $S_d$ for one cut as follows:
\[ S_u = \{(x_i, y_i) \in S | w_r x_r + w_s x_s + b_m \geq 0\}, \]
\[ S_d = \{(x_i, y_i) \in S | w_r x_r + w_s x_s + b_m < 0\}. \]
6: Using an SVM to determine decision functions $f_u(x) = \text{sign}(g_u(x))$ and $f_d(x) = \text{sign}(g_d(x))$ for $S_u$ and $S_d$, respectively.
7: Output the decision function as follows:
\[
 f(x) = \begin{cases} 
 f_u(x), & w_r x_r + w_s x_s + b_m \geq 0, \\
 f_d(x), & w_r x_r + w_s x_s + b_m < 0.
 \end{cases}
\]

In the $n$th stage (final stage), we have two disjoint subsets $T_{i_1i_2...i_{m0}}$ and $T_{i_1i_2...i_{m1}}$. Therefore, we have $2^n$ subsets in $n$th stage. If the cutting hyperplane in the first stage is $w_0 x + b_0 = 0$, then for any $m = 1, 2, \ldots, n - 1$, the cutting hyperplane in the $n$th stage in the subset $T_{i_1i_2...i_m}$ is as follows:
\[
 w_{i_0i_1i_2...i_{m-1}} x + b_{i_0i_1i_2...i_{m-1}} = 0, \quad i_0 = 0. \tag{9}
\]
Also, the decision function for training subsets $T_{i_1i_2...i_n}$ by using the following SVM can be obtained as follows:
\[
 y_j = f_j(x), \quad j = \sum_{k=1}^{n} 2^{i_k}, \tag{10}
\]
where $T_{i_1i_2...i_n}$ is one of the $2^n$ numbers of $T$ in the last stage. Now, the condition $p_{i_1i_2...i_k}(x)$ is defined as follows:
\[
 p_{i_1i_2...i_k}(x) = \begin{cases} 
 w_{i_0i_1i_2...i_{k-1}} x + b_{i_0i_1i_2...i_{k-1}} \geq 0, & i_k = 1 \\
 w_{i_0i_1i_2...i_{k-1}} x + b_{i_0i_1i_2...i_{k-1}} < 0, & i_k = 0.
 \end{cases}
\]
Finally, the decision function will be
\[
\begin{align*}
    f(x) &= \left\{ \begin{array}{ll}
    f_0, & p_0(x), p_{00}(x), \ldots, p_{00\ldots0}(x) \\
    \vdots & \vdots \\
    f_j(x), & (p_i(x), p_{i_1}(x), \ldots, p_{i_1i_2\ldotsi_n}(x)), \quad j = \sum_{k=1}^{n} 2^{i_k} \\
    \vdots & \vdots \\
    f_{2^n-1}, & p_1(x), p_{11}(x), \ldots, p_{11\ldots1}(x) 
\end{array} \right. 
\end{align*}
\]

We test CA for some synthetics data to show the accuracy and performance of the proposed method. For instance, Figure 7 shows the geometry interpretation of solving a classification problem by using CA in two stages and LS-SVM. We denote by “+” the points that belong to the positive class and also we denote by “*” the points that belong to the negative class. The problem is nondominant in the rate 0.5 (i.e., the positive class has 50 vectors and the negative class has 100 vectors). It is clearly seen that, this problem is not separable linearly and applying the linear LS-SVM has less accuracy. However, using the CA in LS-SVM has more accuracy. One can check from Figure 7 that, the CA is useful in two cases: The first one decreases the computational and the second one increases the accuracy. Moreover, Figure 8 depicts the geometric interpretation of nonlinear LS-SVM on nondominant problem by using CA in three-stage cuts. In these synthetics data the LS-SVM cannot do well but applying the CA in LS-SVM the efficient results follow.

![Figure 7: Geometric interpretation of CA in three-stage cuts](image)

**Remark 1.** The aim of this paper is to reduce the computations in LS-SVM and increase the accuracy. By increasing the number of stages in CA, the accuracy is improves but the computations may be increase. For this purpose, increasing the stages in high cases are not recommended.
6 Comparative results

The proposed approach in this paper is investigated by several numerical examples. All computations have been performed with symbolic computation software MATLAB and the calculations are implemented on a machine with Intel core 5 Duo processor 2 GHz and 4 GB RAM. We investigate the performance of the proposed approach and compare with the method without using the CA on some well-known data sets. Table 1 shows the properties of databases such as the number of samples for the corresponding set and the dimensions [11, 2]. We consider $C = 10$ in LS-SVM. Also, for nonlinear LS-SVM, we choose radius basis function kernel with $\sigma = 1$. Tables 2 and 3 indicate the accuracy (percent) of accepted classification with linear LS-SVM and linear LS-SVM with CA, respectively. Now, consider two synthetic datasets that cannot classify with a linear LS-SVM. However, we can use the proposed CA to classify the dataset with linear LS-SVM. This is another advantage of the proposed algorithm. In Figures 9 and 10, the dash line is the LS-SVM classifying the dataset wrongly, but by applying the
Table 3: Accuracy of linear LS-SVM with CA

<table>
<thead>
<tr>
<th>Dataset</th>
<th>BUPA Liver</th>
<th>Heart-Statlog</th>
<th>Sonar</th>
<th>Ionosphere</th>
<th>Australian</th>
<th>CMC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy (%)</td>
<td>71.01</td>
<td>87.04</td>
<td>90.87</td>
<td>92.02</td>
<td>86.52</td>
<td>71.35</td>
</tr>
</tbody>
</table>

Figure 9: The dash-line is LS-SVM which is wrong but the CA classify the dataset correctly.

CA, the dataset is classified in one hand correctly and on the other hand linearly. For more comparison, consider a synthetic dataset which neither the LS-SVM can classify nor TW-SVM (Twin Support Vector Machine). Kumar and Gopal [6] proposed a least squares twin SVM (LS-TW-SVM) for pattern classification. They performed their approach on a synthetic dataset and the result is shown in Figure 11. It shows that their approach is not effective to classify the data. Also, we perform the LS-SVM and the LS-SVM with CA on another synthetic data like the one in [6]. The result is shown in Figure 12.

Figure 10: The dash-line is LS-SVM which is wrong but the CA classify the dataset correctly.
Figure 11: Classification results of LS-T-SVM

Figure 12: Classification results of LS-SVM and LS-SVM with CA

One can see that, the LS-SVM cannot classify the data. However, the LS-SVM with CA does as well. Table 4 shows the comparison of classification accuracy for LS-SVM, LS-TW-SVM with TW-SVM, GEP-SVM (GEP-SVM: generalized eigenvalue proximal SVM [9]), and LS-SMM with CA on 6 UCI datasets. Table 4 shows that the generalization capability of LS-SVM with CA is better than the other methods on many of the datasets considered.
Table 4: Comparison in accuracy for linear kernel

<table>
<thead>
<tr>
<th>Dataset</th>
<th>LS-SVM</th>
<th>LS-T-SVM</th>
<th>T-SVM</th>
<th>GEP-SVM</th>
<th>LS-SVM with CA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bupa Liver</td>
<td>70.43</td>
<td>70.90</td>
<td>70.5</td>
<td>66.36</td>
<td>71.01</td>
</tr>
<tr>
<td>Iheart-statlog</td>
<td>84.81</td>
<td>85.55</td>
<td>86.66</td>
<td>85.55</td>
<td>87.04</td>
</tr>
<tr>
<td>Sonar</td>
<td>87.50</td>
<td>80.47</td>
<td>80.52</td>
<td>79.47</td>
<td>90.87</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>90.03</td>
<td>89.70</td>
<td>88.23</td>
<td>84.11</td>
<td>92.02</td>
</tr>
<tr>
<td>Australian</td>
<td>86.09</td>
<td>86.61</td>
<td>86.91</td>
<td>80.00</td>
<td>86.52</td>
</tr>
<tr>
<td>CMC</td>
<td>68.36</td>
<td>68.84</td>
<td>68.84</td>
<td>68.76</td>
<td>71.35</td>
</tr>
</tbody>
</table>

7 Conclusions

In this paper, a new algorithm for reducing the computations and improving the accuracy of the LS-SVM was given. We called this algorithm as Cutting Algorithm (CA). In that, by using some cuts, we tried to reduce the training stage and therefore reducing the computations. In fact, we broke the original problem into smaller subproblems. By solving the subproblems the original problem was solved. We tested the proposed algorithm on some known datasets. In addition, we showed that the proposed CA can classify the nonlinear datasets, linearly. The reported results showed that the accuracy and the efficiency of the approach. Finally, the work is in progress to extend the approach to solve this problem by neural network models.

References


5. Hao, P.-Y. New support vector algorithms with parametric insensitive/margin model, Neural Networks, 23(1) (2010), 60–73.


