An Improvement of Least Square - Twin Support Vector Machine

Nguyen The Cuong, Nguyen Thanh Vi
Faculty of Basic, Telecommunications University, Khanh Hoa, Vietnam
Tác giả liên hệ: Nguyen The Cuong, nckcbnckcb@gmail.com
Ngày nhận bài: 26/03/2021, ngày sửa chữa: 01/06/2021, ngày duyệt đăng: 12/06/2021
Định danh DOI: 10.32913/mic-ict-research-vn.v2021.n1.970

Abstract: In binary classification problems, two classes of data seem to be different from each other. It is expected to be more complicated due to the number of data points of clusters in each class also be different. Traditional algorithms as Support Vector Machine (SVM), Twin Support Vector Machine (TSVM), or Least Square Twin Support Vector Machine (LSTSVM) cannot sufficiently exploit information about the number of data points in each cluster of the data. Which may be effect to the accuracy of classification problems. In this paper, we propose a new Improvement Least Square - Support Vector Machine (called ILS-SVM) for binary classification problems with a class-vs-cluster strategy. Experimental results show that the ILS-SVM training time is faster than that of TSVM, and the ILS-SVM accuracy is better than LSTSVM and TSVM in most cases.

Keywords: *Support vector machine, twin support vector machine, least square twin support vector machine.*

Tên bài: **Một Cải Tiến Của Máy Véc-tơ Tựa Song Sinh Dùng Bình Phương Tối Thiểu**

Tóm tắt: Trong bài toán phân loại nhị phân, hai lớp dữ liệu thường khác biệt nhau. Phức tạp hơn là số lượng các điểm dữ liệu của mỗi cụm trong mỗi lớp cũng khác nhau. Các thuật toán phân loại truyền thống như Máy Véc-tơ Tựa (SVM), Máy Véc-tơ Tựa Song Sinh (TSVM) hay Máy Véc-tơ Tựa Song Sinh Dùng Bình Phương Tối Thiểu (LSTSVM) không khai thác đầy đủ thông tin về số lượng các điểm trong mỗi cụm. Điều này có thể ảnh hưởng đến độ chính xác của bài toán phân loại. Trong bài này, chúng tôi giới thiệu một thuật toán phân loại nhị phân mới, là cải tiến của LSTSVM (được gọi là ILS-SVM) với chiến lược lớp-vs-cụm. Các kết quả thực nghiệm chỉ ra rằng thời gian huấn luyện của ILS-SVM là nhanh hơn của TSVM, độ chính xác của ILS-SVM là tốt hơn LSTSVM và TSVM trong hầu hết các trường hợp.

Từ khóa: *Máy véc-tơ tựa, máy véc-tơ tựa song sinh, máy véc-tơ tựa song sinh dùng bình phương tối thiểu*

I. INTRODUCTION

In the early years of the 20th century, the Support Vector Machine (SVM) [1, 2] was a popular binary classification algorithm applied to many different fields in practice [3–7]. The SVM seeks a hyperplane separating two classes so that the margin between them is largest. Actual data is often established with different clusters, but SVM does not fully exploit information about the number of data points of the clusters.

Nowadays, with rapid development, datasets are increasing in number and diversifying in structure. This fact requires classification algorithms to guarantee accuracy and improve the speed. Many variants of SVM have been recently proposed to improve the speed and other task of standard SVM [6–10]. Some typical innovations of SVM are Twin Support Vector Machine (TSVM) [11], Structural Twin Support Vector Machine (S-TSVM) [12], and Least Square Twin Support Vector Machine (LSTSVM) [13]. The main idea of TSVM is to seek two hyperplanes such that each hyperplane is closer to one class and far away from the other by solving two Quadratic Programming Problems (QPPs) whose size are smaller than the QPP in SVM. Despite having to solve two QPPs, the speed of TSVM is approximately four times faster than standard SVM. S-TSVM has the same strategy as TSVM. Besides, S-TSVM fully exploits structural information with cluster granularity into learning the model to build a more reasonable classifier. LSTSVM has the same strategy as TSVM and S-TSVM. But, LSTSVM attempts to solve two modified primal problems of TSVM, instead of two dual problems usually solved. LSTSVM shows that the solution of the two modified primal problems reduces to solving just two Systems of Linear Equations (SLEs) as opposed to solving two QPPs in TSVM.

Based on the strategy of S-TSVM and LSTSVM, we propose a new binary classification model: Improvement Least Square - Support Vector Machine (called ILS-SVM) with a class-vs-cluster strategy. Instead of solving two SLEs as in LSTSVM, ILS-SVM will solve $(l+k)$ SLEs, where k and l are the number of clusters in class $\{+\}$ and class $\{-\}$, respectively. This method allows ILS-SVM to effectively improve computation speed and classification accuracy.

The paper is organized as follows. Section 2 briefly introduces the background of SVM, TSVM, and LSTSVM; Section 3 is devoted to a detailed description of ILS-SVM along with the algorithms and discussions; All experimental results are presented in Section 4, together with the comparative evaluation; The conclusion is given in Section 5. All algorithms are settled by version 3.8.3 of Python Programming Language.

II. BACKGROUND

In this section, we first briefly describe the background of SVM, TSVM, and LSTSVM.

1. Problem

Consider a binary classification problem with the dataset, denoted by matrix C, consisting of m points (each point is a row of C) $\mathbf{x}_j^T \in \mathbb{R}^n,\ j = 1, \ldots, m$. We also write $\mathbf{x}_j \in C$ to indicate that $\mathbf{x}_j$ is a row of C. Suppose that $y_j \in \{-1, 1\}$ is the $j-$th data point label. Class $\{+\}$ consists of m_A points denoted by a matrix $A \subset \mathbb{R}^{m_A \times n}$, class $\{-\}$ consists of m_B points denoted by a matrix $B \subset \mathbb{R}^{m_B \times n}$. There are k clusters in class $\{+\}$, whose $i-$th cluster consists of m_{Ai} points and is denoted by matrix $A_i \subset \mathbb{R}^{m_{Ai} \times n}$. Also, there are l clusters in class $\{-\}$, whose $j-$th cluster consists of m_{Bj} points and is denoted by matrix $B_j \subset \mathbb{R}^{m_{Bj} \times n}$.

2. Support Vector Machine

Standard SVM [1] seeks a hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$ separating class $\{+\}$ and class $\{-\}$ such that the margin $\frac{2}{\|\mathbf{w}\|}$ between two classes is largest. However, Standard SVM is only available when the data is linearly separable. In the case when the data is not linearly separable, Soft SVM [1] is recommended with the more loser constraints:

$$\begin{cases} \min\limits_{\mathbf{w},\ b,\ \boldsymbol{\xi}} & c\mathbf{e}^T\boldsymbol{\xi} + \frac{1}{2}\|\mathbf{w}\|_2^2, \\ \text{s.t.} & D(C\mathbf{w} + \mathbf{e}b) + \boldsymbol{\xi} \geq \mathbf{e},\ \boldsymbol{\xi} \geq \mathbf{0}, \end{cases} \tag{1}$$

where $D \in \mathbb{R}^{m \times m}$ is the diagonal matrix with $D_{jj} = y_j;\ \forall j$, $\boldsymbol{\xi} \in \mathbb{R}^m$ is the vector of slack variables, $c \in \mathbb{R}$ is the penalty coefficient, appropriately selected to adjust the role between terms in the objective function, $\mathbf{e} \in \mathbb{R}^m$ is the vector of ones. A new data point $\mathbf{x}$ will be classified in class $\{+\}$ if $\text{sgn}(f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}+b) > 0$ and in class $\{-\}$ if $\text{sgn}(f(\mathbf{x})) < 0$, (see Figure 1.a).

3. Twin Support Vector Machine

Based on the strategy of Multisurface Proximal Support Vector Classification via Generalized Eigenvalues (GEPSVM) [14], the main idea of TSVM [11] for binary classification problem is to seek two hyperplanes:

- $f_+(\mathbf{x})(= \mathbf{w}_+^T\mathbf{x} + b_+) = 0$ is closer to class $\{+\}$ and far away from class $\{-\}$,
- $f_-(\mathbf{x})(= \mathbf{w}_-^T\mathbf{x} + b_-) = 0$ is closer to class $\{-\}$ and far away from class $\{+\}$

by solving two QPPs as follows:

$$\begin{cases} \min\limits_{\mathbf{w}_+, b_+, \boldsymbol{\xi}} & \frac{1}{2}\|A\mathbf{w}_+ + \mathbf{e}_A b_+\|_2^2 + c_1\mathbf{e}_B^T\boldsymbol{\xi}, \\ \text{s.t.} & -(B\mathbf{w}_+ + \mathbf{e}_B b_+) + \boldsymbol{\xi} \geq \mathbf{e}_B,\ \boldsymbol{\xi} \geq \mathbf{0}, \end{cases} \tag{2}$$

$$\begin{cases} \min\limits_{\mathbf{w}_-, b_-, \boldsymbol{\eta}} & \frac{1}{2}\|B\mathbf{w}_- + \mathbf{e}_B b_-\|_2^2 + c_2\mathbf{e}_A^T\boldsymbol{\eta}, \\ \text{s.t.} & (A\mathbf{w}_+ + \mathbf{e}_A b_+) + \boldsymbol{\eta} \geq \mathbf{e}_A,\ \boldsymbol{\eta} \geq \mathbf{0}. \end{cases} \tag{3}$$

Where c_1, c_2 are penalty coefficients to adjust the role between terms in the objective functions, $\mathbf{e}_A \in \mathbb{R}^{m_A}$, $\mathbf{e}_B \in \mathbb{R}^{m_B}$ are vectors of ones, $\boldsymbol{\xi} \in \mathbb{R}^{m_B}$, $\boldsymbol{\eta} \in \mathbb{R}^{m_A}$ are vectors of slack variables. A new data $\mathbf{x}$ is classified into class $\{+\}$ or class $\{-\}$ depending on whether it is closer to the hyperplane $f_+(\mathbf{x}) = 0$ or the hyperplane $f_-(\mathbf{x}) = 0$, (see Figure 1.b). It has been shown in [11] that the training time of TSVM is approximately four times faster than that of SVM.

4. Least Squares Twin Support Vector Machine

LSTSVM [13] modifies the primal problems (2) and (3) of Linear TSVM in a least squares sense, with the inequality constraints replaced by equality constraints as follows:

$$\begin{cases} \min\limits_{\mathbf{w}_+, b_+} & \frac{1}{2}\|A\mathbf{w}_+ + \mathbf{e}_A b_+\|_2^2 + \frac{1}{2}c_1\boldsymbol{\xi}^T\boldsymbol{\xi}, \\ \text{s.t.} & -(B\mathbf{w}_+ + \mathbf{e}_B b_+) + \boldsymbol{\xi} = \mathbf{e}_B, \end{cases} \tag{4}$$

$$\begin{cases} \min\limits_{\mathbf{w}_-, b_-} & \frac{1}{2}\|B\mathbf{w}_- + \mathbf{e}_B b_-\|_2^2 + \frac{1}{2}c_2\boldsymbol{\eta}^T\boldsymbol{\eta}, \\ \text{s.t.} & (A\mathbf{w}_+ + \mathbf{e}_A b_+) + \boldsymbol{\eta} = \mathbf{e}_A. \end{cases} \tag{5}$$

Note also that the QPPs (4) and (5) use the square of $2-$norm of vectors of slack variables $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$ with weights $\frac{c_1}{2}$ and $\frac{c_2}{2}$ instead of $1-$norm with weight c_1 and c_2 as used in (2) and (3), which makes the constraints $\boldsymbol{\xi} \geq \mathbf{0}$ and $\boldsymbol{\eta} \geq \mathbf{0}$ redundant [13]. A new data point is assigned into class $\{+\}$ or $\{-\}$ in the same manner as in TSVM, (see Figure 1.c). It has been shown in [13] that the training time of LSTSVM is faster than that of TSVM (see TABLE 1).

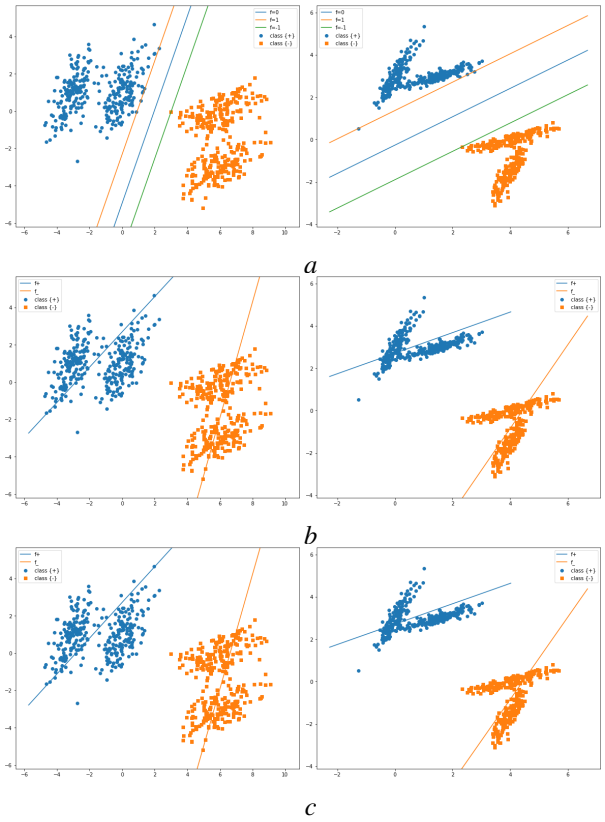

Figure 1. The SVM (a), TSVM (b), and LSTSVM (c) cannot fully exploit information about the number of data-points of clusters in each class.

III. IMPROVEMENT LEAST SQUARE - SUPPORT VECTOR MACHINE

In this section, we describe a new classification algorithm: Improvement Least Square - Support Vector Machine (called ILS-SVM).

Similar to the S-TSVM [12], ILS-SVM also has two steps. The first step is grouping in each class by Ward's linkage clustering method; the second step is the least square model learning. Suppose that, there are k clusters in class $\{+\}$, each cluster consists of m_{Ai} points and is represented by matrix $A_i \subset \mathbb{R}^{m_{Ai}\times n}$, there are l clusters in class $\{-\}$, each cluster consists of m_{Bi} points and is represented by matrix $B_i \subset \mathbb{R}^{m_{Bi}\times n}$. ILS-SVM uses a class-vs-cluster strategy to determine $(l+k)$ hyperplanes such that each of which is closer to one class and far away from cluster of other class. Specifically, the method need to find l hyperplanes such that the $j-$th hyperplane, $f_{j+}(\mathbf{x}) = \mathbf{w}_{j+}^T\mathbf{x} + b_{j+} = 0$, is closer to class A and far away from cluster B_j; Also, It need to find k hyperplanes such that the $i-$th hyperplane, $f_{i-}(\mathbf{x}) = \mathbf{w}_{i-}^T\mathbf{x} + b_{i-} = 0$, is closer to class B and far away from cluster A_i (see Figure 2); Here $\mathbf{w}_{j+}, \mathbf{w}_{i-} \in \mathbb{R}^n, b_{j+}, b_{i-} \in \mathbb{R}$.

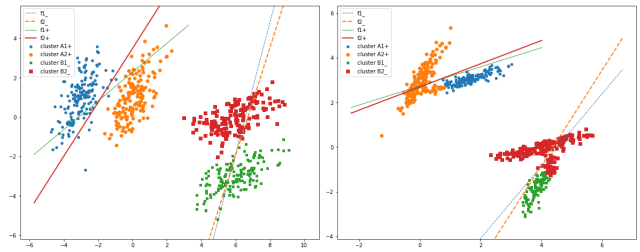

Figure 2. ILS-SVM exploit information about the number of data-points of clusters.

The classifier is now selected as:

$$f(\mathbf{x}) = \underset{+,\ -}{\operatorname{argmin}}(f_+(\mathbf{x}),\ f_-(\mathbf{x})), \tag{6}$$

with

$$f_+(\mathbf{x}) = \sum_{j=1}^{l} \frac{m_{Bj}}{m_B} f_{j+}(\mathbf{x});\ f_-(\mathbf{x}) = \sum_{i=1}^{k} \frac{m_{Ai}}{m_A} f_{i-}(\mathbf{x}). \tag{7}$$

From the definition, we see that $f_+(\mathbf{x})$ is the average, taking into account the weights, distances from $\mathbf{x}$ to the hyperplanes $\{f_{j+}(\mathbf{x}) = 0\}$. The $j-$th hyperplane's weight is proportional to m_{Bj} - the number of data points in the cluster Bj. Similarly, $f_-(\mathbf{x})$ is the weighted average of distances from $\mathbf{x}$ to the hyperplanes $\{f_{i-}(\mathbf{x}) = 0\}$. By virtue of (6), a new data point $\mathbf{x}$ is classified into class $\{+\}$ or $\{-\}$ depending on whether $f_+(\mathbf{x})$ is less than or greater than $f_-(\mathbf{x})$.

1. The linear case

We determine $(l+k)$ hyperplanes in ILS-SVM by solving $(l+k)$ QPPs as follows:

$$\begin{cases} \min\limits_{\mathbf{w}_{j+},b_{j+},\boldsymbol{\xi}_j} & \frac{1}{2}||A\mathbf{w}_{j+}+\mathbf{e}_A b_{j+}||_2^2+\frac{c_1}{2}\boldsymbol{\xi}_j^T\boldsymbol{\xi}_j+\frac{c_2}{2}(\|\mathbf{w}_{j+}\|_2^2+b_{j+}^2), \\ \text{s.t.} & (B_j\mathbf{w}_{j+}+\mathbf{e}_{Bj}b_{j+})+\boldsymbol{\xi}_j = \mathbf{e}_{Bj}, \end{cases} \tag{8}$$

$j = 1, \ldots,\ l$ and

$$\begin{cases} \min\limits_{\mathbf{w}_{i-},b_{i-},\boldsymbol{\eta}_i} & \frac{1}{2}||B\mathbf{w}_{i-}+\mathbf{e}_B b_{i-}||_2^2+\frac{c_3}{2}\boldsymbol{\eta}_i^T\boldsymbol{\eta}_i+\frac{c_4}{2}(\|\mathbf{w}_{i-}\|_2^2+b_{i-}^2), \\ \text{s.t.} & (A_i\mathbf{w}_{i-}+\mathbf{e}_{Ai}b_{i-})+\boldsymbol{\eta}_i = \mathbf{e}_{Ai}, \end{cases} \tag{9}$$

$i = 1, \ldots,\ k$.

Here, $\mathbf{e}_{Ai} \in \mathbb{R}^{m_{Ai}\times 1}$, $\mathbf{e}_{Bj} \in \mathbb{R}^{m_{Bj}\times 1}$, $\mathbf{e}_A \in \mathbb{R}^{m_A\times 1}$, $\mathbf{e}_B \in \mathbb{R}^{m_B\times 1}$ are vectors of ones. $\boldsymbol{\eta}_i \in \mathbb{R}^{m_{Ai}\times 1}$, $\boldsymbol{\xi}_j \in \mathbb{R}^{m_{Bj}\times 1}$ are vectors of slack variables. c_1, c_2, c_3, c_4 are penalty coefficients to adjust the role between terms in the objective functions. In the problem (8), $||A\mathbf{w}_{j+} + \mathbf{e}_A b_{j+}||_2^2$ is the sum of squares of distances from data points in class A to the hyperplane $f_{j+}(\mathbf{x}) = 0$, the inequality constraints are replaced by equality constraints, and is defined by the points of cluster B_j, making training times faster than that of TSVM. $\frac{1}{2}c_1\boldsymbol{\xi}_j^T\boldsymbol{\xi}_j$ is the sum of squares of errors,

$\frac{1}{2}c_2(\|\mathbf{w}_{j+}\|_2^2 + b_{j+}^2)$ is the regularization term. The problem (9) is similarly established for class $\{-\}$ with the constraints is defined by cluster A_i.

By substituting the equality constraints into the objective function, QPP (8) becomes:

$$\min_{\mathbf{w}_{j+},b_{j+}} \frac{1}{2}\|A\mathbf{w}_{j+} + \mathbf{e}_A b_{j+}\|_2^2 + \frac{c_1}{2}\|B_j\mathbf{w}_{j+} + \mathbf{e}_{Bj}b_{j+} - \mathbf{e}_{Bj}\|_2^2 + \frac{c_2}{2}(\|\mathbf{w}_{j+}\|_2^2 + b_{j+}^2). \tag{10}$$

Setting the gradient of (10) with respect to $\mathbf{w}_{j+}$ and b_{j+} to zero, we have:

$$A^T(A\mathbf{w}_{j+}+\mathbf{e}_A b_{j+})+c_1B_j^T(B_j\mathbf{w}_{j+}+\mathbf{e}_{Bj}b_{j+}-\mathbf{e}_{Bj})+c_2\mathbf{w}_{j+} = \mathbf{0},$$

$$\mathbf{e}_A^T(A\mathbf{w}_{j+}+\mathbf{e}_A b_{j+})+c_1\mathbf{e}_{Bj}^T(B_j\mathbf{w}_{j+}+\mathbf{e}_{Bj}b_{j+}-\mathbf{e}_{Bj})+c_2 b_{j+} = 0.$$

Defining $H = \begin{bmatrix}A & \mathbf{e}_A\end{bmatrix}$, $G_j = \begin{bmatrix}B_j & \mathbf{e}_{Bj}\end{bmatrix}$, $\mathbf{u}_j = \begin{bmatrix}\mathbf{w}_{j+}\\ b_{j+}\end{bmatrix}$, $j = 1,\ldots,l$, and I being the identity matrix of order $(n+1)$, it follows that

$$H^THu_j + c_1G_j^TG_j\mathbf{u}_j - c_1G_j^T\mathbf{e}_{Bj} + c_2I\mathbf{u}_j = \mathbf{0} \tag{11}$$

$$\Rightarrow \left[\tfrac{1}{c_1}H^TH + G_j^TG_j + \tfrac{c_2}{c_1}I\right]\mathbf{u}_j = G_j^T\mathbf{e}_{Bj} \tag{12}$$

$$\Rightarrow \mathbf{u}_j = \left[\tfrac{1}{c_1}H^TH + G_j^TG_j + \tfrac{c_2}{c_1}I\right]^{-1}G_j^T\mathbf{e}_{Bj}. \tag{13}$$

In exactly similar way, by defining $G = \begin{bmatrix}B & \mathbf{e}_B\end{bmatrix}$, $H_i = \begin{bmatrix}A_i & \mathbf{e}_{Ai}\end{bmatrix}$, $\mathbf{v}_i = \begin{bmatrix}\mathbf{w}_{i-}\\ b_{i-}\end{bmatrix}$, $i = 1,\ldots,k$, we obtain the solutions of problem (9):

$$\mathbf{v}_i = \left[\tfrac{1}{c_3}G^TG + H_i^TH_i + \tfrac{c_4}{c_3}I\right]^{-1}H_i^T\mathbf{e}_{Ai}. \tag{14}$$

Algorithm 1: Linear ILS-SVM

1 Consider the problem 2.1. We generate the linear classifier $f(\mathbf{x})$ as follows:
2 Clustering dataset by using Ward's linkage [12].
3 Define $H_i = \begin{bmatrix}A_i & \mathbf{e}_{Ai}\end{bmatrix}$, $G = \begin{bmatrix}B & \mathbf{e}_B\end{bmatrix}$, $G_j = \begin{bmatrix}B_j & \mathbf{e}_{Bj}\end{bmatrix}$, $H = \begin{bmatrix}A & \mathbf{e}_A\end{bmatrix}$.
4 Determining $(\mathbf{w}_{j+}, b_{j+})$ and $(\mathbf{w}_{i-}, b_{i-})$ via (13), (14).
5 Classifying a new data $\mathbf{x}$ by using (6) and (7).

2. The nonlinear case

Let $\Phi : \mathbb{R}^n \to \mathbb{H}$ be a nonlinear mapping, where $\mathbb{H}$ is a Hilbert space whose dimension is not less than n (maybe infinite-dimensional). Since $\mathbb{S} = \text{span}(\Phi(C^T))$ is a subspace of $\mathbb{H}$ whose dimension does not exceed m, we can consider $\mathbb{S}$ as an Euclidean space and $\Phi : \mathbb{R}^n \to \mathbb{S}$. Suppose that after the clustering step on space $\mathbb{S}$ we obtain k clusters $\Phi(A_1),\ldots,\Phi(A_k)$ in the class $\Phi(A)$, each cluster $\Phi(A_i)$ consists of m_{Ai} data points; and l clusters $\Phi(B_1),\ldots,\Phi(B_l)$ in the class $\Phi(B)$, each cluster $\Phi(B_j)$ consists of m_{Bj} data points. In space $\mathbb{S}$, a hyperplane $\Phi(\mathbf{x}^T)\mathbf{h} + b = 0$ (with $\mathbf{h} \in \mathbb{S}$ being the normal vector) can be rewritten as $\Phi(\mathbf{x}^T)\Phi(C^T)\mathbf{u} + b = 0$ for some vector $\mathbf{u} \in \mathbb{R}^m$. Therefore, by defining $\Phi(\mathbf{x}^T)\Phi(C^T) = K(\mathbf{x}^T, C^T)$, the hyperplane has the form $K(\mathbf{x}^T, C^T)\mathbf{u} + b = 0$, K is a predefined kernel [15].

ILS-SVM determines l hyperplanes such that the $j-$th one: $K(\mathbf{x}^T,\ C^T)\mathbf{u}_{j+}+b_{j+} = 0$ is closer to class $\Phi(A)$ and far away from cluster $\Phi(B_j)$. It also determines k hyperplanes such that the $i-$th one: $K(\mathbf{x}^T,\ C^T)\mathbf{u}_{i-}+b_{i-} = 0$ is closer to class $\Phi(B)$ and far away from cluster $\Phi(A_i)$. Specifically, we have $(l + k)$ QPPs as follows:

$$\begin{cases}\min\limits_{\mathbf{u}_{j+},b_{j+},\boldsymbol{\xi}_j} \frac{1}{2}\|K(A,C^T)\mathbf{u}_{j+}+\mathbf{e}_A b_{j+}\|_2^2+\frac{c_1}{2}\boldsymbol{\xi}_j^T\boldsymbol{\xi}_j+\frac{c_2}{2}(\|\mathbf{u}_{j+}\|_2^2+b_{j+}^2),\\ \text{s.t.} \quad (K(B_j,C^T)\mathbf{u}_{j+} + \mathbf{e}_{Bj}b_{j+}) + \boldsymbol{\xi}_j = \mathbf{e}_{Bj};\end{cases} \tag{15}$$

$\mathbf{u}_{j+} \in \mathbb{R}^m$, $j = 1,\ldots,l$ and

$$\begin{cases}\min\limits_{\mathbf{u}_{i-},b_{i-},\boldsymbol{\eta}_i} \frac{1}{2}\|K(B,C^T)\mathbf{u}_{i-}+\mathbf{e}_B b_{i-}\|_2^2+\frac{c_3}{2}\boldsymbol{\eta}_i^T\boldsymbol{\eta}_i+\frac{c_4}{2}(\|\mathbf{u}_{i-}\|_2^2+b_{i-}^2),\\ \text{s.t.} \quad (K(A_i,C^T)\mathbf{u}_{i-} + \mathbf{e}_{Ai}b_{i-}) + \boldsymbol{\eta}_i = \mathbf{e}_{Ai};\end{cases} \tag{16}$$

$\mathbf{u}_{i-} \in \mathbb{R}^m$, $i = 1,\ldots,k$.

By substituting the constraints into objective function, these QPPs become:

$$\min_{\mathbf{u}_{j+},b_{j+}} \frac{1}{2}\|K(A,C^T)\mathbf{u}_{j+} + \mathbf{e}_A b_{j+}\|_2^2 + \frac{c_1}{2}\|K(B_j,C^T)\mathbf{u}_{j+} + \mathbf{e}_{Bj}b_{j+} - \mathbf{e}_{Bj}\|_2^2 + \frac{c_2}{2}(\|\mathbf{u}_{j+}\|_2^2 + b_{j+}^2), \tag{17}$$

$j = 1,\ldots,l$ and

$$\min_{\mathbf{u}_{i-},b_{i-}} \frac{1}{2}\|K(B,C^T)\mathbf{u}_{i-} + \mathbf{e}_B b_{i-}\|_2^2 + \frac{c_3}{2}\|K(A_i,C^T)\mathbf{u}_{i-} + \mathbf{e}_{Ai}b_{i-} - \mathbf{e}_{Ai}\|_2^2 + \frac{c_4}{2}(\|\mathbf{u}_{i-}\|_2^2 + b_{i-}^2), \tag{18}$$

$i = 1,\ldots,k$.

The solution of QPPs (17) and (18) can be derived to be:

$$\overline{\mathbf{u}_j} = \left[\tfrac{1}{c_1}H^TH + G_j^TG_j + \tfrac{c_2}{c_1}I\right]^{-1}G_j^T\mathbf{e}_{Bj}, \tag{19}$$

$$\overline{\mathbf{v}_i} = \left[\tfrac{1}{c_3}G^TG + H_i^TH_i + \tfrac{c_4}{c_3}I\right]^{-1}H_i^T\mathbf{e}_{Ai}, \tag{20}$$

where, $H_i = \begin{bmatrix}K(A_i,C^T) & \mathbf{e}_{Ai}\end{bmatrix}$, $G = \begin{bmatrix}K(B,C^T) & \mathbf{e}_B\end{bmatrix}$, $G_j = \begin{bmatrix}K(B_j,C^T) & \mathbf{e}_{Bj}\end{bmatrix}$, $H = \begin{bmatrix}K(A,C^T) & \mathbf{e}_A\end{bmatrix}$, $\overline{\mathbf{u}_j} = \begin{bmatrix}\mathbf{u}_{j+}\\ b_{j+}\end{bmatrix}$, $j = \overline{1,l}$, $\overline{\mathbf{v}_i} = \begin{bmatrix}\mathbf{u}_{i-}\\ b_{i-}\end{bmatrix}$, $i = \overline{1,k}$, and I is the identity matrix of order $(m + 1)$.

The classification function now is selected as:

$$f(\mathbf{x}) = \underset{+,\ -}{\text{argmin}}(f_+(\mathbf{x}),\ f_-(\mathbf{x})), \tag{21}$$

with

$$\begin{cases} f_+(\mathbf{x}) = \sum_{j=1}^{l} \frac{m_{Bj}}{m_B}(K(\mathbf{x}^T, C^T)\mathbf{u}_{j+} + b_{j+}); \\ f_-(\mathbf{x}) = \sum_{i=1}^{k} \frac{m_{Ai}}{m_A}(K(\mathbf{x}^T, C^T)\mathbf{u}_{i-} + b_{i-}). \end{cases} \quad (22)$$

Remark: In (19), (20) we need to compute the inverse of square matrices in order $(m + 1)$. This work will become difficult when m is large. So it is necessary to reduce the size of those matrices. This problem can be solved by using the Sherman-Morrison-Woodbury (SMW) formula [16].

Algorithm 2: Nonlinear ILS-SVM

1 Consider the problem 2.1. We generate the nonlinear classifier $f(\mathbf{x})$ as follows:
2 Choosing a kernel function $K(\mathbf{x}^T, C^T)$, typically the Gaussian kernel [15].
3 Clustering the dataset by using Ward's linkage [12].
4 Define $H_i = \left[K(A_i, C^T) \quad \mathbf{e}_{Ai}\right]$, $G = \left[K(B, C^T) \quad \mathbf{e}_B\right]$, $G_j = \left[K(B_j, C^T) \quad \mathbf{e}_{Bj}\right]$, $H = \left[K(A, C^T) \quad \mathbf{e}_A\right]$.
5 Determining $(\mathbf{u}_{1+}, b_{1+}), \ldots, (\mathbf{u}_{l+}, b_{l+})$ and $(\mathbf{u}_{1-}, b_{1-}), \ldots, (\mathbf{u}_{k-}, b_{k-})$ via (19) and (20).
6 Classifying a new data $\mathbf{x}$ by using (21) and (22).

IV. EXPERIMENTS

In this section, we compare the ILS-SVM against LSTSVM [13] and TSVM [11] on various datasets. All algorithms are settled by version 3.8.3 of Python programming language, and run on a Laptop with an AMD Ryzen 5 with 8GB RAM. We use the following libraries: "scipy.cluster.hierarchy" to cluster the data, "cvxopt" to solve the QPP, "matplotlib" to show the figures, "pandas" and "numpy" to process data, 'sklearn' to evaluate and adjust hyperparameters of all models. All settings are uploaded to [17]. We implement these algorithms on 14 UCI datasets [18] which have been experimented in [11], [13]. For each dataset, we randomly select 90% of extracted dataset for training and 10% for testing, and use 10-fold cross validation (CV) to evaluate the accuracy of all algorithms. The results are shown in TABLE 1 (by applying Algorithm 1), and TABLE 2 (by applying Algorithm 2).

TABLE 1 shows the training time and the classification accuracy comparison between ILS-SVM with LSTSVM and TSVM for the linear case on 14 UCI datasets. All hyperparameters such as c_1, c_2, c_3, c_4 of ILS-SVM, c_1, c_2 of LSTSVM and TSVM are belong to the set $\{0.0001, 0.001, 0.1, 1, 10, 100, 1000\}$ and obtained by using grid-search technique. From TABLE 1 we can see that the training time of ILS-SVM is faster than that of TSVM, but slower than LSTSVM. This is because ILS-SVM has to cluster the data and process the problem on each cluster. That is the cost to obtain a more precise classification. Thereby we can see that the classification accuracy of ILS-SVM is better than that of LSTSVM and TSVM in most datasets. The TABLE 2 shows the comparison of training time and classification accuracy for the nonlinear version of ILS-SVM, LSTSVM and TSVM on 4 UCI datasets. The kernel parameters of the Gaussian kernel are obtained through grid-search from the same range, while all penalty parameters of all algorithms are fixed to be one.

Table I
TEST TRAINING TIME (MS) AND CV ACCURACY (%) WITH A LINEAR KERNEL (ALG. 1)

Datasets($m \times n$) ($k \times l$)	Algorithms		
	ILS-SVM	LSTSVM	TSVM
Hepatitis(155×19) (5×3)	3.001 89.9 +/- 6.6	1.001 86.4 +/- 11.7	8.002 84.2 +/- 8.9
Australian(690×14) (4×5)	6.001 87.3 +/- 4.6	1.000 86.3 +/- 3.6	57.014 86.8 +/- 3.4
BUPA-liver(345×6) (2×3)	3.017 67.4 +/- 6.7	1.000 66.1 +/- 7.8	14.003 65.8 +/- 6.0
CMC(844×9) (3×5)	8.989 66.5 +/- 5.8	1.000 64.6 +/- 5.1	143.032 64.8 +/- 5.2
Credit(690×19) (2×3)	12.001 86.5 +/- 2.0	2.000 86.3 +/- 2.7	103.022 86.1 +/- 3.3
Diabetes(768×8) (5×2)	4.981 74.9 +/- 5.0	0.001 74.3 +/- 4.6	24.006 74.7 +/- 5.7
Flare-sol(1066×10) (2×2)	19.985 82.1 +/- 4.5	0.999 81.9 +/- 3.6	244.055 81.8 +/- 4.3
German(1000×20) (3×2)	16.986 71.7 +/- 6.2	1.000 70.8 +/- 5.5	221.048 71.0 +/- 7.2
Heart-stat(270×13) (2×2)	2.000 84.0 +/- 5.5	0.001 84.8 +/- 3.8	10.001 84.0 +/- 3.9
Image(2310×18) (3×2)	6.999 90.2 +/- 1.0	1.000 90.2 +/- 3.0	51.011 91.6 +/- 2.2
Ionosphere(350×34) (3×2)	5.005 89.2 +/- 7.5	0.999 89.6 +/- 6.2	15.002 88.0 +/- 6.9
Spect(265×22) (3×3)	3.987 86.1 +/- 4.7	0.001 83.5 +/- 6.7	9.995 83.1 +/- 6.1
Sonar(208×60) (6×3)	3.993 81.3 +/- 10.3	1.000 74.3 +/- 9.5	9.002 73.8 +/- 9.6
Heart-c(303×13) (2×4)	2.000 85.3 +/- 7.1	0.001 83.8 +/- 6.3	12.003 83.4 +/- 7.7

Table II
TEST TRAINING TIME (S) AND CV ACCURACY (%) WITH AN RBF KERNEL (ALG. 2)

Dataset($m \times n$) ($k \times l$)	Algorithms		
	ILS-SVM	LSTSVM	TSVM
Hepatitis(155×19) (5×3)	0.271 87.0 +/- 9.1	0.119 84.9 +/- 9.3	0.125 83.5 +/- 12.0
heart-c(303×13) (2×4)	0.931 83.1 +/- 7.3	0.460 81.6 +/- 6.0	0.472 82.7 +/- 7.1
Australian(690×14) (4×5)	4.911 87.6 +/- 4.2	2.377 81.8 +/- 3.2	2.446 85.6 +/- 3.8
WPBC(198×35) (2×3)	0.374 81.7 +/- 9.3	0.183 78.1 +/- 6.3	0.194 80.6 +/- 10.2

V. CONCLUSION

This paper has proposed a new Improvement Least Square - Support Vector Machine for classification problems with a class-vs-cluster strategy. This algorithm is performed in two steps: The first step is grouping in each class by Ward's linkage clustering method; The second step is the least square learning model. The classifier is based on the weighted average distances from the data point to the class representative hyperplanes. ILS-SVM has a faster execution time than TSVM, and slower than LSTSVM. But in terms of classification accuracy, ILS-SVM is better than TSVM, and LSTSVM in most cases. For problems with big data and each class contains many clusters, the ILS-SVM algorithm is more effective in data classification. The ILS-SVM algorithm may not really be suitable for multi-classes problems. However, it seems useful in solving the classification problem with imbalanced data. And this is an interesting research direction in the future.

REFERENCES

[1] V. Vapnik, *The Natural Of Statistical Learning Theory*. Springer-Verlag New York, 1995.

[2] G. Fung and O. L. Mangasarian, "Proximal support vector machine," in *KDD '01: Proceedings of the Seventh ACM SIGKDD International Conference On Knowledge Discovery And Data Mining*. San Francisco California: Association for Computing Machinery, New York, NY, United States, 2001, pp. 77–86. [Online]. Available: https://dl.acm.org/doi/10.1145/502512.502527

[3] W. Noble, *Support Vector Machine Applications in Computational Biology*. MIT Press, 2004.

[4] M. Adancon and M. Cheriet, "Model selection for the ls-svm. application to handwriting recognition," *Pattern Recognition*, vol. 42, pp. 3264–3270, 2009.

[5] Y. Tian, Y. Shi, and Y. Liu, "Recent advances on support vector machines research," *Technological and Economic Development of Economy*, vol. 18, pp. 5–33, 2012.

[6] D. Tomar and S. Agarwal, "Twin support vector machine: A review from 2007 to 2014," *Egyptian Informatics Journal*, vol. 16, pp. 55–69, 2015.

[7] J. Cervantes, F. Lamont, L. Mazahua, and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing*, vol. 408, pp. 189–215, 2020.

[8] X. Pan, Y. Luo, and Y. Xu, "K-nearest neighbor based structural twin support vector machine," *Knowledge-Based Systems*, vol. 88, pp. 34–44, 2015.

[9] X. Xie and S. Sun, "Multitask centroid twin support vector machines," *Neurocomputing*, vol. 149, pp. 1085–1091, 2015.

[10] B. Mei and Y. Xu, "Multi-task least squares twin support vector machine for classification," *Neurocomputing*, vol. 338, pp. 26–33, 2019.

[11] Jayadeva, R. Khemchandani, and S. Chandra, "Twin support vector machines for pattern classification," *IEEE Transactions on Pattern Analysis and Machine intelligence*, vol. 29, pp. 905–910, 2007.

[12] Z. Qi, Y. Tian, and Y. Shi, "Structural twin support vector machine for classification," *Knowledge-Based Systems*, vol. 43, pp. 74–81, 2013.

[13] M. Kumar and M. Gopal, "Least squares twin support vector machines for pattern classification," *Expert Systems with Applications*, vol. 36, pp. 7535–7543, 2009.

[14] O. Mangasarian and E. Wild, "Multisurface proximal support vector classification via generalized eigenvalues," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, pp. 69–74, 2006.

[15] B. Schoelkopf and A. Smola, *Learning with Kernel*. MIT Press, 2002.

[16] G. Golub and C. Van Loan, *Matrix Computations*. The John Hopkins University Press, 2013.

[17] N. Cuong, *Python code*. [Online]. Available: https://github.com/makeho8/python/

[18] *UCI Machine Learning Repository*, Center for Machine Learning and Intelligent Systems at the University of California, Irvine. [Online]. Available: http://archive.ics.uci.edu/ml/machine-learning-databases/

Cuong Nguyen The received his B.Sc. degree in Mathematical Education from Hanoi National University of Education, Vietnam in 2009. M.Sc. degree in Mathematical analysis from Hue University College of Sciences, Vietnam in 2014. He is currently working toward his Ph.D. degree in Computer Science at Hue University College of Sciences. He works at Telecommunications University.
Email: nckcbnckcb@gmail.com.

Vi Nguyen Thanh received her B.Sc. degree in Mathematical Education from Da Nang University, Vietnam in 2011. M.Sc. degree in Mathematical analysis from Hue University College of Sciences, Vietnam in 2014. She works at Telecommunications University.
Email: thanhvy.rose@gmail.com