Pattern Recognition 63 (2017) 139–148

Contents lists available at ScienceDirect

Pattern Recognition

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

Robust support vector machines based on the rescaled hinge loss function

Guibiao Xu[a,*], Zheng Cao[b], Bao-Gang Hu[a], Jose C. Principe[b]

[a] NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[b] CNEL, Department of Electrical & Computer Engineering, University of Florida, Gainesville, FL 32611, USA

ARTICLE INFO

Keywords:
Support vector machine
Robustness
Rescaled hinge loss
Half-quadratic optimization

ABSTRACT

The support vector machine (*SVM*) is a popular classifier in machine learning, but it is not robust to outliers. In this paper, based on the Correntropy induced loss function, we propose the rescaled hinge loss function which is a monotonic, bounded and nonconvex loss that is robust to outliers. We further show that the hinge loss is a special case of the proposed rescaled hinge loss. Then, we develop a new robust SVM based on the rescaled hinge loss. After using the half-quadratic optimization method, we find that the new robust SVM is equivalent to an iterative weighted SVM, which can help explain the robustness of iterative weighted SVM from a loss function perspective. Experimental results confirm that the new robust SVM not only performs better than SVM and the existing robust SVMs on the datasets that have outliers, but also presents better sparseness than SVM.

1. Introduction

The support vector machine (*SVM*) is one of the most popular classifiers which not only holds good theoretical foundation but also achieves significant empirical success in various applications [1,2]. It is known that SVM can be fit in the regularization framework of *loss* + *penalty* using the hinge loss function $l_{hinge}(z) = \max(0, 1 - z)$, where z is the margin variable [3]. In the regularization framework, the loss term is in charge of the fidelity of the resulting model to training samples and the *penalty* term is used to avoid overfitting. In practical applications, training samples are often contaminated by noise and some even have wrong labels [4]. These are usually known as outliers. In this paper, we consider outliers as the samples that locate in the other classes and are far away from their own classes due to feature noise or label noise [4,5]. Unfortunately, in such cases, researchers have shown that SVM is not robust to outliers [6]. The main reason is that the unboundedness of $l_{hinge}(z)$ causes outliers to have large losses and as a result, the decision boundary of SVM may deviate severely from the optimal one, which results in poor classification performance [5,7,8]. Besides, outliers also make the sparseness of SVM much worse [5,7]. In order to mitigate the effects of outliers, different approaches have been proposed to improve the robustness of SVM.

Song et al. [7] suggested to use the distance between each training sample and its class center to calculate an adaptive margin so as to reduce the influence of outliers. Weighted SVM (*WSVM*)[1] was also proposed to deal with outliers [8–10]. In WSVM, different weights are assigned to different training samples which can show their importance in the training dataset. Several weight functions have been proposed [8–10] among which some are based on the distance between a training sample and its class center, and some are based on the distance between a training sample and the decision boundary of SVM. But, most of the existing weight functions are heuristic. Ding and Xu [11] presented a novel combinatorial technique, which was called random gradient descent (*RGD*) tree, to identify and remove outliers in SVM and developed a new algorithm called RGD-SVM. Moreover, some researchers suggested to develop a robust SVM by applying the bounded ramp loss $l_{ramp}(z) = \max(0, 1 - z) - \max(0, r - z)$, where $r \le 0$ is a given constant that determines the location of truncation [5,12–16]. Since $l_{ramp}(z)$ places bounded losses on outliers, it does not overfocus on outliers and thus, is robust to outliers. Besides, it can also improves the sparseness of the resulting classifier [5,12,14]. The resulting optimization problem of $l_{ramp}(z)$ is nonconvex and different kinds of optimization methods have been proposed to solve it. Wu and Liu [5] applied difference of convex function programming which was also known as concave-convex procedure [12]. Xu et al. [13] proposed semi-definite programming along with convex relaxation. Besides, online learning [14] and linear programming [15] were also suggested to optimize the nonconvex problem of $l_{ramp}(z)$. Suzumura et al. [16] developed outlier-path algorithm which can also control the robustness of the resulting model during the optimization. The experimental results in the above papers show that the above approaches can improve the robustness of SVM to outliers.

In this paper, we propose a novel loss function called the rescaled hinge loss function, which is then used to develop a new robust SVM by

* Corresponding author.
E-mail addresses: xuguibiao@gmail.com (G. Xu), zcaozcao@gmail.com (Z. Cao), hubg@nlpr.ia.ac.cn (B.-G. Hu), principe@cnel.ufl.edu (J.C. Principe).

[1] In some papers, WSVM is also called fuzzy SVM [9,10].

http://dx.doi.org/10.1016/j.patcog.2016.09.045
Received 30 September 2015; Received in revised form 7 March 2016; Accepted 26 September 2016
Available online 29 September 2016

utilizing the half-quadratic (*HQ*) optimization method [17]. The main contribution of this work is as follows

- Inspired by the Correntropy-induced loss function (*C-loss*) [18], we propose the rescaled hinge loss which is a monotonic, bounded and nonconvex loss. It is robust to outliers. And moreover, we further show that $l_{hinge}(z)$ is a special case of the rescaled hinge loss.
- We develop a new robust SVM based on the rescaled hinge loss, which is equivalent to an iterative WSVM after using HQ optimization method. As far as we know, it is the first time that the relationship between an iterative WSVM and a loss function is established. And this new relationship helps explain the robustness of iterative WSVM from a loss function perspective.

Compared with the optimization problem of $l_{ramp}(z)$, it is easy to implement our robust SVM with the help of LIBSVM [19]. And experimental results confirm that the new robust SVM is effective in improving the robustness of SVM to outliers and its sparseness.

This paper is structured as follows. In Section 2, we introduce the rescaled hinge loss. The development of the new robust SVM based on the rescaled hinge loss as well as its relationship with iterative WSVM are presented in Section 3. Section 4 shows the experimental results and we summarize the paper in Section 5.

2. Rescaled hinge loss function

Correntropy is an information theoretic metric [20] and has been commonly used in robust learning [21]. For example, under the framework of sparse representation classifier [22], the sparse representation classifiers based on Correntropy are proposed for robust face recognition in [23,24]. In [18], Singh et al. used Correntropy to do classification and developed the C-loss

$$l_C(z) = \beta\left[1 - \exp\left(-\frac{(1-z)^2}{2\sigma^2}\right)\right], \tag{1}$$

where σ is window width and $\beta = [1 - \exp(-1/(2\sigma^2))]^{-1}$ is a normalizing constant which ensures that $l_C(0) = 1$. C-loss embeds higher order statistics of z, and has been used to train neural network [18] and kernel adaptive classifier [25]. The least square (*LS*) loss function is defined as $l_{ls}(z) = (1-z)^2$ [26], which is a popular convex loss function. Then, we can rewrite $l_C(z)$ as follows:

$$l_C(z) = \beta[1 - \exp(-\eta l_{ls}(z))], \tag{2}$$

where $\eta = 1/(2\sigma^2) > 0$ is viewed as a scaling constant and $\beta = 1/(1 - \exp(-\eta))$. In other words, we can explain $l_C(z)$ as the rescaled LS loss. Fig. 1(a) shows $l_{ls}(z)$ and $l_C(z)$ with different η's, and we can see that after rescaling, C-loss becomes a bounded, smooth and nonconvex loss. However, like the LS loss, C-loss is not monotonic either. According to the relationship between $l_C(z)$ and $l_{ls}(z)$, we derive the rescaled hinge loss in a similar way

$$l_{rhinge}(z) = \beta[1 - \exp(-\eta l_{hinge}(z))], \tag{3}$$

where $\beta = 1/(1 - \exp(-\eta))$ is a normalizing constant which also ensures that $l_{rhinge}(0) = 1$. Fig. 1(b) shows $l_{hinge}(z)$ and $l_{rhinge}(z)$ with different η's, and we can see that after rescaling, the unbounded $l_{hinge}(z)$ becomes the bounded $l_{rhinge}(z)$. Meanwhile, different from C-loss, $l_{rhinge}(z)$ is monotonic. We can also learn from Fig. 1(b) that $\eta(\eta > 0)$ controls the upper bound of $l_{rhinge}(z)$. Proposition 1 states that $l_{rhinge}(z)$ becomes $l_{hinge}(z)$ as $\eta \to 0$.

Proposition 1. $\lim_{\eta\to 0} l_{rhinge}(z) = l_{hinge}(z)$.

Proof. According to the Taylor series of the exponential function, we know that

$$l_{rhinge}(z) = \sum_{j=1}^{+\infty} \frac{\beta\eta^j(-1)^{j+1} l_{hinge}^j(z)}{j!}.$$

According to L'Hopital's rule, we derive that

$$\lim_{\eta\to 0}\beta\eta^j = \lim_{\eta\to 0}\frac{\eta^j}{1-\exp(-\eta)} = \begin{cases}1 & \text{if } j = 1,\\ 0 & \text{if } j \geq 2.\end{cases}$$

Consequently, $\lim_{\eta\to 0} l_{rhinge}(z) = l_{hinge}(z)$.□We show by Proposition 1 that $l_{hinge}(z)$ can be viewed as a special case of $l_{rhinge}(z)$. In the next section, we derive the new robust SVM based on $l_{rhinge}(z)$.

3. Robust SVM based on $l_{rhinge}(z)$

We consider the following linear discriminant function:

$$f(\boldsymbol{x}) = \boldsymbol{w}^T\phi(\boldsymbol{x}) + b, \tag{4}$$

Table 1
Experimental datasets.

	Dataset	# of Classes	# of Samples	# of Features	Class ratio
D1	Australian	2	690	14	1: 1.25
D2	Pima	2	768	8	1: 1.87
D3	German Credit	2	1000	24	1: 2.33
D4	Spambase	2	4601	57	1: 1.54
D5	a5a	2	6414	122	1: 3.09
D6	Musk	2	6598	166	1: 5.49
D7	Svmguide1	2	7089	4	1: 1.29
D8	Twonorm	2	7400	20	1: 1.00
D9	Protein (1 VS. 2)	2	9568	357	1:1.35
D10	Magic Gamma	2	19 020	10	1: 1.84

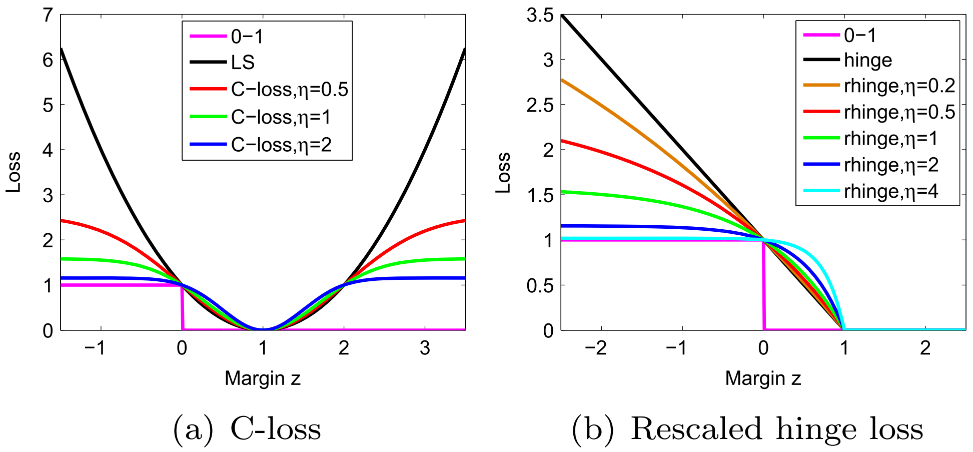

Fig. 1. Different loss functions for classification.

Table 2
The values of $R_1(\boldsymbol{w}, b)$ across the iterations of HQ optimization method. The Gaussian kernel is used, $\gamma = 1/d$ and $C = 1$.

Iteration		1	2	⋯	10	⋯	29	30	ϱ (%)
Australian	$\eta = 0.5$	180.24	174.90	⋯	172.92	⋯	172.87	172.87	99.33
	$\eta = 1$	177.10	162.14	⋯	157.94	⋯	157.94	157.94	100.00
	$\eta = 2$	189.73	157.32	⋯	141.72	⋯	141.60	141.60	99.76
Pima	$\eta = 0.5$	334.62	325.13	⋯	323.94	⋯	323.94	323.94	99.99
	$\eta = 1$	329.98	303.65	⋯	294.77	⋯	294.40	294.40	98.95
	$\eta = 2$	344.12	282.51	⋯	267.46	⋯	266.25	266.25	98.45
German	$\eta = 0.5$	434.39	419.76	⋯	418.47	⋯	418.40	418.40	99.51
	$\eta = 1$	442.72	401.17	⋯	393.70	⋯	392.66	392.66	97.91
	$\eta = 2$	491.52	390.20	⋯	364.45	⋯	361.05	360.95	97.32
Spambase	$\eta = 0.5$	817.80	794.26	⋯	791.44	⋯	791.44	791.44	99.98
	$\eta = 1$	818.81	754.34	⋯	745.03	⋯	745.00	745.00	99.97
	$\eta = 2$	873.75	738.43	⋯	709.14	⋯	709.13	709.08	99.99
a5a	$\eta = 0.5$	1975.58	1901.44	⋯	1873.83	⋯	1872.25	1872.25	98.48
	$\eta = 1$	1888.13	1758.27	⋯	1646.56	⋯	1643.90	1643.90	98.91
	$\eta = 2$	1841.08	1615.53	⋯	1422.07	⋯	1409.19	1409.19	97.02
Musk	$\eta = 0.5$	563.81	549.91	⋯	549.09	⋯	549.08	549.08	99.99
	$\eta = 1$	585.63	541.58	⋯	535.29	⋯	535.26	535.26	99.96
	$\eta = 2$	577.15	538.78	⋯	531.37	⋯	531.35	531.35	100.00
Svmguide1	$\eta = 0.5$	606.53	599.76	⋯	598.98	⋯	598.98	598.98	99.99
	$\eta = 1$	595.33	578.12	⋯	575.90	⋯	575.86	575.86	99.75
	$\eta = 2$	604.06	571.85	⋯	562.63	⋯	562.29	562.29	99.20
Twonorm	$\eta = 0.5$	421.00	412.55	⋯	412.09	⋯	412.09	412.09	99.99
	$\eta = 1$	431.27	405.42	⋯	402.27	⋯	402.24	402.23	99.88
	$\eta = 2$	479.52	404.51	⋯	394.21	⋯	393.63	393.61	99.29
Protein	$\eta = 0.5$	3268.44	3183.32	⋯	3174.94	⋯	3174.91	3174.91	99.99
	$\eta = 1$	3463.13	3218.28	⋯	3146.03	⋯	3145.16	3145.13	99.72
	$\eta = 2$	3879.18	3471.87	⋯	3181.46	⋯	3167.83	3167.79	98.08
Magic	$\eta = 0.5$	5553.17	5358.94	⋯	5334.61	⋯	5334.60	5334.60	100.00
	$\eta = 1$	5322.84	4790.82	⋯	4696.12	⋯	4695.61	4695.60	99.92
	$\eta = 2$	5309.00	4291.41	⋯	4137.05	⋯	4133.87	4133.85	99.73

where w is the weight vector; b is the bias term and $\phi(\boldsymbol{x})$ is the feature mapping function which is usually defined by a kernel function that satisfies the Mercer theory [1]. We denote the training dataset as $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ is a feature vector (d is feature dimension) and $y_i\{\in \pm 1\}$ is a binary class label. According to the regularization framework of *loss* + *penalty* [3], we consider the following objective function:

$$\min_{\boldsymbol{w},b} R_1(\boldsymbol{w}, b) = \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^N l_{rhinge}(z_i), \tag{5}$$

where $z_i = y_i f(\boldsymbol{x}_i)$ and $C \ge 0$ is the regularization parameter. By simple arithmetic modification, (5) is equivalent to the following problem:

$$\max_{\boldsymbol{w},b} R_2(\boldsymbol{w}, b) = -\frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\beta\sum_{i=1}^N \exp(-\eta l_{hinge}(z_i)). \tag{6}$$

We recognize that HQ optimization method [17] can be used here to optimize (6), which has been successfully applied in robust face recognition [23,24] and robust feature extraction [27]. Notice that the methods in [23,24] are sparse representation classifiers based on Correntropy, which are proposed for robust face recognition, and the method in [27] is based on Renyi's entropy, which is proposed for robust feature extraction. However, our proposed method is a discriminant classifier based on $l_{rhinge}(z)$, which is proposed to improve the robustness of SVM to outliers. In the next section, we derive HQ optimization method for (6).

3.1. HQ optimization for $l_{rhinge}(z)$

We define a convex function $g(v) = -v\log(-v) + v$, where $v < 0$. According to the conjugate function theory [28], we have

$$\exp(-\eta l_{hinge}(z)) = \sup_{v<0}\{\eta l_{hinge}(z)v - g(v)\}, \tag{7}$$

where the supremum is achieved at $v = -\exp(-\eta l_{hinge}(z)) < 0$. We put the derivation of (7) in Appendix A. Using (7), we rewrite $R_2(\boldsymbol{w}, b)$ in (6) as

$$\begin{aligned} R_2(\boldsymbol{w}, b) &= -\frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\beta\sum_{i=1}^N \sup_{v_i<0}\{\eta l_{hinge}(z_i)v_i - g(v_i)\} = -\frac{1}{2}\|\boldsymbol{w}\|_2^2 \\ &+ C\beta \sup_{\boldsymbol{v}<\boldsymbol{0}}\{\sum_{i=1}^N (\eta l_{hinge}(z_i)v_i - g(v_i))\} \\ &= \sup_{\boldsymbol{v}<\boldsymbol{0}}\{-\frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\beta\sum_{i=1}^N (\eta l_{hinge}(z_i)v_i - g(v_i))\}, \end{aligned} \tag{8}$$

where $\boldsymbol{v} \in \mathbb{R}^N$ and $\boldsymbol{v}<\boldsymbol{0}$. In (8), the second equation establishes since $\eta l_{hinge}(z_i)v_i - g(v_i)$, $i = 1, 2, \ldots, N$ are independent functions in terms of v_i, and the third equation establishes since $-1/2\|\boldsymbol{w}\|_2^2$ is a constant with respect to v_i. With (8), we can derive that (6) is equivalent to

$$\max_{\boldsymbol{w},b,\boldsymbol{v}<\boldsymbol{0}} R_3(\boldsymbol{w}, b, \boldsymbol{v}) = -\frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\beta\sum_{i=1}^N (\eta l_{hinge}(z_i)v_i - g(v_i)). \tag{9}$$

After having (9), we can use the alternating optimization method to optimize (9). To be specific, given $(\boldsymbol{w}, b)$, we optimize over $\boldsymbol{v}$ and then

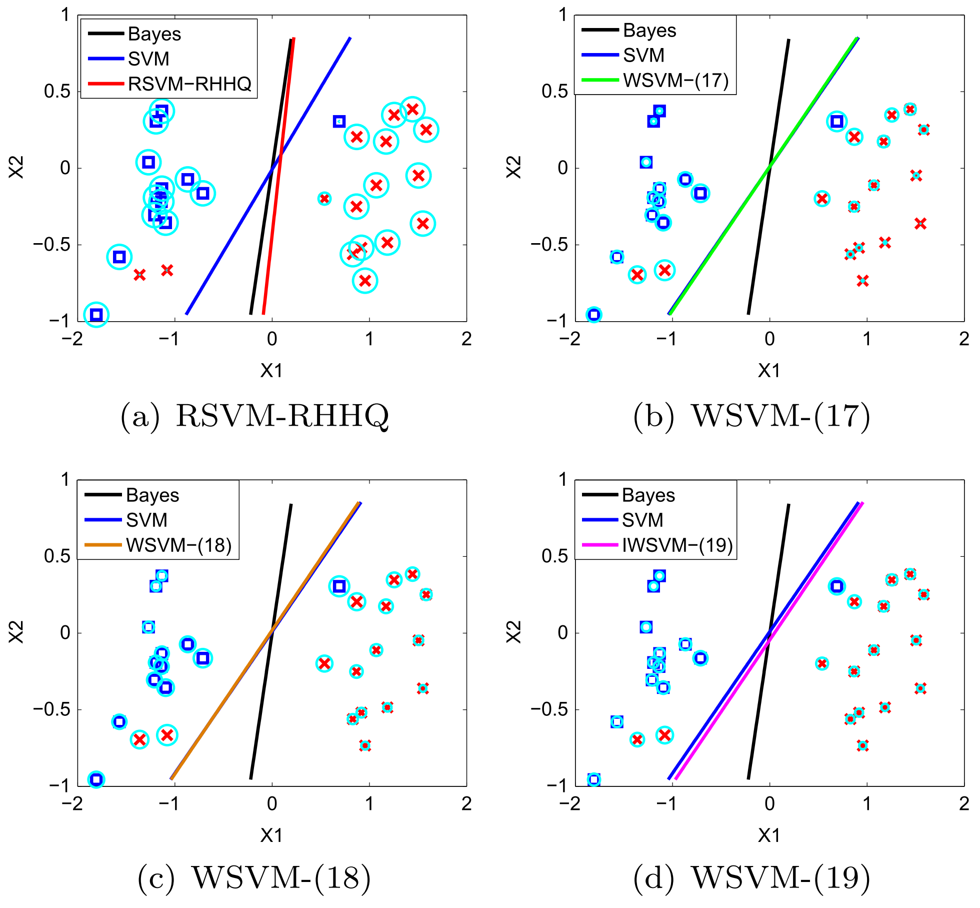

Fig. 2. The linear decision boundaries of RSVM-RHHQ, WSVM- (17), WSVM- (18) and WSVM- (19) on an artificial dataset with outliers. The radii of circles denote the magnitude of ω_i.

Table 3
The loss bounds of $l_{rhinge}(z)$ with different η's.

η	0.2	0.5	1	2	3
Loss bound	5.52	2.54	1.58	1.16	1.02

given $\boldsymbol{v}$, we optimize over $(\boldsymbol{w}, b)$. First, suppose that we are given $(\boldsymbol{w}^s, b^s)$ (the superscript s denotes the s-th iteration), then (9) is equivalent to

$$\max_{\boldsymbol{v}^s<\boldsymbol{0}} \sum_{i=1}^{N} (\eta l_{hinge}(z_i^s)v_i^s - g(v_i^s)), \tag{10}$$

whose analytic solutions are

$$v_i^s = -\exp(-\eta l_{hinge}(z_i^s)) < 0, \quad i = 1, 2, \ldots, N. \tag{11}$$

Second, after we obtain $\boldsymbol{v}^s$, we can get $(\boldsymbol{w}^{s+1}, b^{s+1})$ by solving the following problem:[2]

$$\max_{\boldsymbol{w},b} \ -\frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\beta\sum_{i=1}^{N} \eta l_{hinge}(z_i)v_i, \tag{12}$$

whose equivalent problem is

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i=1}^{N} C_i l_{hinge}(z_i), \tag{13}$$

where $C_i = C\beta\eta(-v_i) \geq 0$. We can see that (13) is a WSVM problem which has already been well studied [8–10]. Using the Lagrange multipliers, (13) can be converted into an equivalent dual problem [5,6,13]

$$\min_{\boldsymbol{\alpha}} \ \frac{1}{2}\boldsymbol{\alpha}^T K\boldsymbol{\alpha} - \boldsymbol{1}^T\boldsymbol{\alpha}\text{s. t.} \quad \boldsymbol{y}^T\boldsymbol{\alpha} = 0, \quad 0 \leq \alpha_i \leq C_i, \quad i = 1, 2, \ldots, N, \tag{14}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^N$ are Lagrange multipliers and $\boldsymbol{\alpha}\geqslant\boldsymbol{0}$; $K_{ij} = y_i y_j \phi^T(\boldsymbol{x}_i)\phi(\boldsymbol{x}_j) = y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, where $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is a kernel function; and $\boldsymbol{1}$ is a vector of all ones. In this paper, we use LIBSVM [19] to help solve (14). Once the solution of (14) is obtained, $\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \phi(\boldsymbol{x}_i)$ and b can be computed using the KKT conditions. The training samples with $\alpha_i > 0$ are called support vectors. After solving (14), the linear discriminant function (4) becomes

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i y_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b. \tag{15}$$

Now, we have HQ optimization method for (5) whose key steps are (11) and (14). Algorithm 1 presents the whole optimization procedure. In Algorithm 1, $\boldsymbol{v}^0$ can be initialized by either setting it to $-\boldsymbol{1}$ or using some existing weight functions [9,10].

Algorithm 1. HQ optimization method for (5).

```
Input:
  The training dataset $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$; $\eta$ in $l_{rhinge}(z)$; the regularization
parameter $C$; the kernel function $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$; and the iteration number
$S$.
  Output: $\boldsymbol{\alpha}$ and $b$ in (15).
1: Construct $K$ using $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$;
2: Set $s$:=0 and initialize $\boldsymbol{v}^s$;
3: while $s < S$ do
4: Obtain $(\boldsymbol{\alpha}^{s+1}, b^{s+1})$ by solving (14);
5: Update $\boldsymbol{v}^{s+1}$ according to (11) and (15);
```

[2] We omit the superscripts of $(\boldsymbol{w}^{s+1}, b^{s+1})$, z_i^{s+1} and v_i^s for the reason of clarity.

Table 4
The 10-fold cross validation accuracies of all the classifiers with linear kernel. The best results are highlighted in bold (mean(std)%).

(a) 0% label noise

Dataset	SVM	WSVM- (19)	RSVM-Ramp	RSVM-RHHQ					
				$\eta = 0.2$	$\eta = 0.5$	$\eta = 1$	$\eta = 2$	$\eta = 3$	CV on η
D1	85.38(4.90)	84.94(3.99)	85.51(3.93)	85.66(4.34)	85.67(4.76)	**86.10(4.34)**	85.09(4.70)	85.09(4.61)	85.67(4.76)
D2	77.10(6.88)	77.35(7.36)	77.22(7.10)	77.35(6.77)	**78.00(6.24)**	77.87(5.66)	77.61(5.78)	77.74(6.47)	77.87(6.58)
D3	74.60(4.48)	75.50(4.25)	75.40(4.43)	**75.60(4.84)**	75.40(4.14)	75.40(4.17)	75.30(3.97)	75.20(4.16)	75.60(5.64)
D4	92.85(1.58)	93.20(1.48)	92.74(1.40)	93.26(1.46)	93.22(1.39)	93.26(1.36)	93.04(1.38)	92.87(1.38)	**93.30(1.49)**
D5	84.14(1.02)	84.08(0.98)	83.93(1.34)	**84.18(0.92)**	83.74(0.98)	83.86(1.20)	83.91(1.16)	84.05(1.24)	84.02(0.69)
D6	95.29(0.53)	95.57(0.59)	95.53(0.43)	95.57(0.50)	95.71(0.39)	95.71(0.59)	95.67(0.72)	95.62(0.62)	**95.73(0.51)**
D7	95.63(0.65)	95.95(0.74)	96.05(0.81)	96.04(0.80)	**96.11(0.76)**	96.05(0.83)	95.95(0.86)	95.87(0.87)	96.09(0.76)
D8	97.73(0.40)	97.73(0.47)	97.76(0.53)	97.77(0.42)	97.77(0.41)	97.73(0.45)	97.77(0.40)	**97.78(0.36)**	97.77(0.40)
D9	80.15(1.16)	80.02(1.24)	**80.42(1.36)**	80.38(1.05)	80.33(1.15)	80.27(1.55)	80.26(1.59)	80.19(1.26)	80.39(1.21)
D10	79.14(0.50)	79.42(0.52)	**79.98(0.78)**	79.82(0.48)	79.71(0.50)	79.94(0.58)	79.83(0.52)	79.70(0.44)	79.71(0.50)

(b) 15% label noise

Dataset	SVM	WSVM- (19)	RSVM-Ramp	RSVM-RHHQ					
				$\eta = 0.2$	$\eta = 0.5$	$\eta = 1$	$\eta = 2$	$\eta = 3$	CV on η
D1	85.52(4.52)	85.52(4.52)	85.23(4.99)	85.52(4.52)	85.52(4.52)	85.52(4.52)	85.52(4.52)	85.52(4.52)	85.52(4.52)
D2	76.18(5.66)	76.70(6.50)	76.57(6.09)	76.96(5.69)	76.96(6.10)	**77.74(5.62)**	77.22(5.04)	77.09(5.87)	77.09(6.29)
D3	73.80(2.62)	74.10(3.90)	73.80(3.36)	74.00(3.20)	**75.00(3.33)**	74.60(3.37)	74.60(3.81)	74.00(2.31)	74.70(3.83)
D4	88.78(2.01)	90.78(1.74)	91.61(0.95)	90.46(1.73)	91.74(1.15)	92.15(0.91)	92.13(1.06)	91.89(1.11)	**92.26(1.06)**
D5	82.49(1.32)	83.21(1.36)	83.04(1.29)	82.43(1.39)	82.69(2.05)	82.85(1.61)	83.19(1.04)	**83.22(1.04)**	82.91(1.69)
D6	93.48(0.52)	93.97(0.67)	93.86(0.85)	93.62(0.68)	93.80(0.80)	94.26(0.84)	**94.27(0.62)**	94.18(0.75)	94.18(0.85)
D7	92.41(1.21)	94.75(0.89)	94.85(0.60)	94.06(1.06)	95.70(0.67)	95.78(0.89)	95.37(0.92)	95.25(0.93)	**95.82(0.84)**
D8	97.61(0.47)	97.66(0.50)	97.74(0.57)	97.64(0.53)	97.68(0.36)	97.73(0.37)	**97.82(0.35)**	97.77(0.46)	97.78(0.39)
D9	78.43(1.51)	78.69(1.02)	78.49(1.54)	78.67(1.33)	78.50(1.84)	78.57(1.71)	78.84(1.25)	78.84(1.20)	**79.01(1.99)**
D10	78.27(0.63)	79.06(0.53)	**79.95(0.89)**	78.76(0.59)	79.27(0.53)	79.86(0.48)	79.78(0.53)	79.94(0.53)	79.86(0.48)

(c) 30% label noise

Dataset	SVM	WSVM- (19)	RSVM-Ramp	RSVM-RHHQ					
				$\eta = 0.2$	$\eta = 0.5$	$\eta = 1$	$\eta = 2$	$\eta = 3$	CV on η
D1	83.61(4.52)	83.61(4.52)	83.32(3.65)	83.61(4.52)	83.61(4.52)	83.61(4.52)	83.61(4.52)	83.61(4.52)	83.61(4.52)
D2	72.55(5.68)	73.19(5.49)	72.94(6.11)	72.55(6.31)	73.46(6.68)	72.94(5.73)	73.19(5.39)	72.93(5.74)	**73.59(6.12)**
D3	69.50(3.50)	70.10(3.28)	68.90(5.00)	69.80(2.57)	70.70(2.71)	71.50(3.69)	71.50(3.78)	71.80(4.85)	**71.90(4.79)**
D4	84.26(2.02)	86.57(2.53)	88.20(2.18)	85.35(2.35)	87.52(2.61)	**89.39(2.53)**	88.52(2.15)	87.70(2.32)	89.22(2.41)
D5	76.77(1.12)	76.99(1.29)	**78.25(2.89)**	76.55(1.50)	76.40(1.60)	76.54(1.64)	77.13(1.95)	77.83(1.17)	77.13(1.95)
D6	92.20(1.19)	92.45(1.11)	92.60(0.99)	92.53(1.12)	92.67(1.09)	92.92(1.31)	**93.07(1.26)**	93.04(1.01)	92.94(1.34)
D7	87.33(2.64)	92.31(1.00)	93.40(1.15)	88.59(2.16)	93.88(1.25)	**95.50(0.71)**	94.23(1.09)	93.51(1.26)	95.40(0.75)
D8	97.62(0.31)	97.65(0.32)	97.12(0.73)	97.69(0.31)	97.69(0.45)	97.58(0.48)	**97.73(0.41)**	97.69(0.45)	97.70(0.40)
D9	72.30(1.47)	71.27(1.64)	72.16(1.41)	72.16(1.65)	72.29(1.81)	72.26(1.91)	72.37(1.95)	72.47(1.76)	**72.89(2.02)**
D10	76.61(0.81)	77.95(0.54)	79.71(0.81)	77.48(0.68)	78.56(0.61)	**79.80(0.59)**	79.29(0.61)	79.61(0.61)	79.65(0.71)

```
6: Set $s := s + 1$;
7: end while
8: return $\boldsymbol{\alpha} = \boldsymbol{\alpha}^s$ and $b = b^s$.
```

Proposition 2 states that Algorithm 1 converges.

Proposition 2. *The sequence* $\{R_{C3}(\boldsymbol{w}^s, b^s, \boldsymbol{v}^s), s = 1, 2, \cdots\}$ *generated by Algorithm 1 converges.*

Proof. According to (8), we know that $R_{C3}(\boldsymbol{w}, b, \boldsymbol{v}) \le R_{C2}(\boldsymbol{w}, b) \le C\beta N$, i.e., $R_{C3}(\boldsymbol{w}, b, \boldsymbol{v})$ is upper bounded. Then, we conclude from (10) and (12) that

$$R_{C3}(\boldsymbol{w}^s, b^s, \boldsymbol{v}^s) \le R_{C3}(\boldsymbol{w}^{s+1}, b^{s+1}, \boldsymbol{v}^s) \le R_{C3}(\boldsymbol{w}^{s+1}, b^{s+1}, \boldsymbol{v}^{s+1}),$$

which means that the sequence $\{R_{C3}(\boldsymbol{w}^s, b^s, \boldsymbol{v}^s), s = 1, 2, \cdots\}$ generated by Algorithm 1 is nondecreasing. Consequently, the sequence $\{R_{C3}(\boldsymbol{w}^s, b^s, \boldsymbol{v}^s), s = 1, 2, \cdots\}$ *of* Algorithm 1 converges.□

We call the new robust SVM, which is based on $l_{rhinge}(z)$ and optimized by HQ optimization method, *RSVM-RHHQ* for short.

3.2. RSVM-RHHQ and Iterative WSVM

According to the derivation in Section 3.1, we find that the proposed RSVM-RHHQ is equivalent to an iterative WSVM. At the s-th iteration, we learn from (11) that the weight function for the next iteration is

$$\omega_i = -v_i^s = \exp(-\eta l_{hinge}(y_i f^s(\boldsymbol{x}_i))), \tag{16}$$

where ω_i denotes the weight of $\boldsymbol{x}_i$. There are some existing weight functions for WSVMs which are based on the output $f(\boldsymbol{x})$ of SVM [8,10]

$$\omega_i = 1 - \frac{|f(\boldsymbol{x}_i)|}{\max(|f(\boldsymbol{x}_i)|) + \Delta}, \tag{17}$$

$$\omega_i = \frac{2}{1 + \exp(\eta |f(\boldsymbol{x}_i)|)}, \tag{18}$$

Table 5
The 10-fold cross validation accuracies of all the classifiers with the Gaussian kernel. The best results are highlighted in bold (mean(std)%).

(a) 0% label noise

Dataset	SVM	WSVM- (19)	RSVM-Ramp	RSVM-RHHQ					
				$\eta = 0.2$	$\eta = 0.5$	$\eta = 1$	$\eta = 2$	$\eta = 3$	CV on η
D1	84.50(4.79)	85.80(4.41)	**86.54(4.64)**	86.09(3.54)	85.95(3.90)	85.66(4.05)	85.81(4.87)	85.52(4.62)	86.09(3.54)
D2	76.83(6.78)	77.22(6.74)	76.84(6.36)	77.35(5.55)	77.22(6.83)	77.09(5.92)	77.22(6.93)	**77.61(6.16)**	77.35(5.55)
D3	75.80(3.08)	75.30(3.09)	75.00(3.71)	76.20(4.34)	**76.90(4.70)**	76.10(3.75)	76.20(4.08)	76.20(5.14)	76.40(3.44)
D4	**93.46(1.11)**	93.15(1.00)	91.67(1.21)	93.28(1.20)	93.20(1.26)	92.91(1.25)	92.74(1.09)	92.78(1.26)	93.28(1.20)
D5	83.33(1.11)	83.50(0.92)	**83.93(1.08)**	83.35(0.85)	83.16(1.09)	83.29(1.29)	83.16(0.93)	83.13(0.93)	83.35(0.85)
D6	**99.03(0.45)**	98.71(0.44)	91.76(0.86)	99.00(0.44)	98.83(0.35)	98.38(0.44)	97.91(0.52)	97.45(0.59)	99.00(0.44)
D7	96.98(0.71)	96.90(0.78)	96.67(0.71)	97.08(0.71)	97.02(0.74)	96.97(0.70)	96.87(0.72)	96.91(0.74)	**97.09(0.70)**
D8	97.81(0.43)	97.82(0.48)	97.70(0.41)	**97.89(0.36)**	97.88(0.29)	97.85(0.23)	97.80(0.33)	97.77(0.34)	97.86(0.37)
D9	83.50(1.47)	83.33(1.47)	80.51(0.02)	83.42(1.63)	83.51(1.62)	83.63(1.46)	83.34(1.59)	83.37(1.36)	**83.69(1.37)**
D10	87.16(0.62)	87.17(0.71)	83.39(0.48)	**87.23(0.52)**	87.21(0.60)	87.18(0.66)	87.17(0.67)	87.16(0.59)	87.19(0.56)

(b) 15% label noise

Dataset	SVM	WSVM- (19)	RSVM-Ramp	RSVM-RHHQ					
				$\eta = 0.2$	$\eta = 0.5$	$\eta = 1$	$\eta = 2$	$\eta = 3$	CV on η
D1	83.92(5.35)	84.36(4.28)	85.37(4.46)	**85.95(3.12)**	85.81(5.07)	85.81(5.08)	85.81(4.48)	85.81(4.28)	85.66(5.24)
D2	76.45(7.84)	76.44(5.78)	77.09(5.21)	77.48(5.99)	77.48(6.36)	77.59(5.83)	77.83(7.00)	**77.85(5.99)**	77.60(6.37)
D3	73.90(3.51)	74.70(3.65)	74.20(3.22)	74.70(3.47)	75.20(3.77)	75.30(3.66)	**75.70(3.71)**	75.60(4.08)	75.60(3.83)
D4	91.61(1.15)	91.91(1.25)	91.59(1.21)	92.28(0.91)	92.50(1.04)	92.28(1.16)	92.46(0.94)	92.39(1.16)	**92.52(1.06)**
D5	82.06(1.04)	82.79(1.00)	**83.15(1.03)**	82.52(0.95)	82.10(1.13)	82.35(1.25)	82.98(1.47)	82.68(1.18)	82.16(1.04)
D6	96.32(0.90)	96.48(0.79)	90.53(0.98)	**96.91(0.83)**	96.64(0.99)	96.30(0.67)	96.89(0.49)	96.70(0.37)	96.64(0.99)
D7	96.57(0.89)	96.61(0.87)	96.61(0.77)	96.67(0.83)	96.77(0.80)	96.77(0.85)	96.73(0.90)	96.77(0.86)	**96.81(0.90)**
D8	97.24(0.62)	97.31(0.71)	97.57(0.47)	97.70(0.55)	97.73(0.49)	97.78(0.47)	**97.81(0.37)**	97.77(0.38)	97.78(0.47)
D9	79.94(1.18)	80.39(1.38)	77.51(1.65)	80.71(1.55)	80.81(1.49)	80.52(1.21)	80.78(1.12)	80.85(1.41)	**80.90(1.51)**
D10	85.88(0.59)	**86.25(0.57)**	82.93(0.52)	85.77(0.57)	85.86(0.56)	86.19(0.51)	86.01(0.53)	86.00(0.53)	86.21(0.51)

(c) 30% label noise

Dataset	SVM	WSVM- (19)	RSVM-Ramp	RSVM-RHHQ					
				$\eta = 0.2$	$\eta = 0.5$	$\eta = 1$	$\eta = 2$	$\eta = 3$	CV on η
D1	80.58(6.26)	80.73(5.23)	**85.38(4.68)**	84.49(4.11)	84.64(3.55)	84.35(3.61)	84.49(3.56)	84.49(3.62)	84.34(3.64)
D2	71.49(6.23)	72.92(6.01)	73.19(6.12)	73.83(4.86)	72.27(5.53)	73.57(5.58)	73.97(5.12)	**74.10(4.95)**	73.97(5.30)
D3	70.70(3.95)	70.40(3.81)	70.10(1.52)	71.50(3.24)	71.00(4.27)	**71.60(4.48)**	71.55(4.92)	71.58(5.60)	71.60(5.13)
D4	88.98(2.07)	89.04(2.47)	88.28(2.66)	89.48(2.03)	89.96(2.24)	90.18(1.86)	90.02(1.41)	90.09(1.16)	**90.39(1.71)**
D5	80.14(1.35)	80.59(2.02)	**81.95(1.20)**	81.07(1.24)	80.74(1.58)	81.32(1.00)	81.67(1.20)	81.71(0.99)	81.45(1.12)
D6	93.42(0.76)	93.62(0.97)	84.65(0.15)	94.21(0.68)	93.98(0.95)	94.09(1.08)	94.26(1.08)	94.27(0.92)	**94.29(0.94)**
D7	95.80(0.54)	96.23(0.68)	96.33(0.78)	96.06(0.60)	96.26(0.59)	96.43(0.56)	**96.43(0.71)**	96.16(0.63)	96.18(0.73)
D8	96.08(0.62)	95.84(0.80)	97.27(0.41)	97.49(0.53)	97.49(0.54)	97.64(0.39)	97.70(0.29)	**97.74(0.30)**	97.69(0.31)
D9	73.87(1.27)	73.23(1.42)	71.51(1.56)	76.32(1.26)	76.87(1.55)	76.54(1.74)	76.96(4.75)	76.90(5.57)	**76.97(1.74)**
D10	83.03(0.64)	83.52(0.73)	**84.64(0.64)**	82.73(0.67)	82.15(0.99)	82.85(0.99)	83.76(0.71)	83.42(0.70)	83.69(0.69)

$$\omega_i = \frac{1}{1 + |f(\mathbf{x}_i)|}, \tag{19}$$

where Δ is a small positive constant. As a general rule, we hope to place small ω_i's on outliers so that their influence is reduced. The weight functions (17)–(19) are based on the distances between training samples and SVM's decision boundary. All of them implicitly assume that outliers locate farther from the decision boundary of SVM than normal training samples. However, this assumption is intuitive and may not establish in some cases. In the experiments, we will present a toy example where (17)–(19) become invalid. Our weight function (16), which is derived from $l_{rhinge}(z_i)$, is based on $l_{hinge}(z_i)$ and for any training sample with large $l_{hinge}(z_i)$, we give it small ω_i in order to reduce its detrimental influence. We think that our motivation is more rational. Also, since there is no guarantee for convergence using (17)–(19) [8,10], the authors suggest that the weights should be updated only once. Meanwhile, for our weight function (16), Proposition 2 ensures its convergence property.

Iterative WSVM and WSVM are known as robust SVMs which is intuitive, but the authors do not give a solid explanation about its robustness in their papers [8–10]. According to the derivation in Section 3.1, we know that the robustness of RSVM-RHHQ, which is also an iterative WSVM, comes from the fact that it is a good approximation to the solution of (5) which is based on $l_{rhinge}(z)$ that is robust to outliers. To the best of our knowledge, it is the first time that the robustness of an iterative WSVM is explained from a loss function perspective.

4. Experiments

In this section, we carry out experiments to show the performance of RSVM-RHHQ. Table 1 shows the experimental datasets which come from [19,29]. As class distributions in each dataset are balanced, we only use *accuracy*, which is the percentage of correctly classified samples, to measure the classification results. Label noise can be viewed as a type of outliers, which is the process that pollutes labels [4]. Like [5,8,11,12,14–16], we also introduce label noise in order to study the robustness of RSVM-RHHQ.

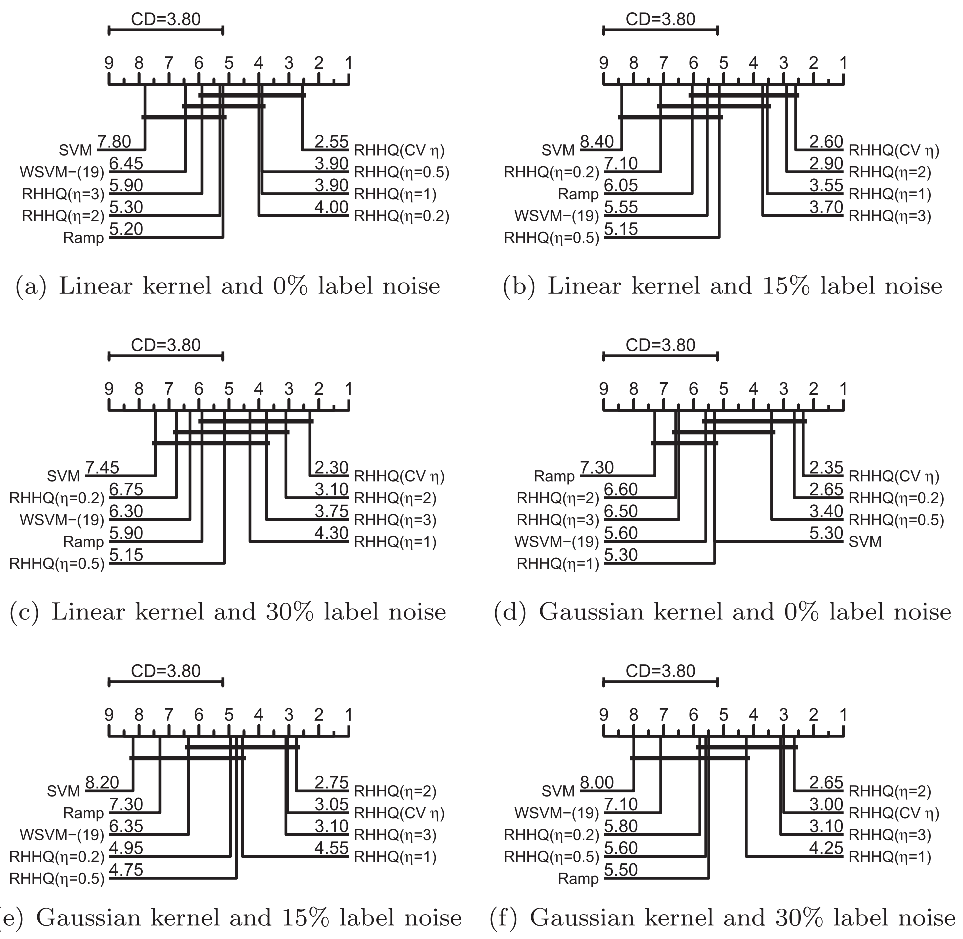

Fig. 3. The CD diagrams of the Friedman test with Nemenyi post-hoc test at the 5% significance level. Groups of classifiers that are not significantly different are connected. Ramp represents RSVM-Ramp. RHHQ ($\eta = 0.2$) represents RSVM-RHHQ with $\eta = 0.2$ and the others have similar meaning with RHHQ ($\eta = 0.2$).

Table 6
The mean support vector ratios of all the classifiers over all the datasets (mean %).

Case	SVM	WSVM- (19)	RSVM-Ramp	RSVM-RHHQ					
				$\eta = 0.2$	$\eta = 0.5$	$\eta = 1$	$\eta = 2$	$\eta = 3$	CV on η
Linear, 0%	31.44	28.93	23.31	27.63	21.94	16.59	16.28	16.49	25.14
Linear, 15%	59.33	54.75	43.51	52.07	41.92	31.38	28.88	29.08	35.95
Linear, 30%	79.81	77.08	60.58	74.51	63.59	48.06	43.15	42.47	50.60
Gaussian, 0%	36.02	37.26	40.95	34.12	31.80	33.49	30.14	27.99	33.52
Gaussian, 15%	63.47	62.88	53.46	59.57	53.02	44.73	39.88	40.08	49.16
Gaussian, 30%	80.79	82.52	64.99	74.85	67.30	58.43	49.56	47.91	54.11

4.1. Convergence of HQ optimization method

In this section, we experimentally study the convergence behavior of HQ optimization method so that we have an idea about how to set S in Algorithm 1. The Gaussian kernel $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\gamma\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2)$ is used, $\gamma = 1/d$, $C = 1$ and $S = 30$. Table 2 shows the values of $R_1(\boldsymbol{w}, b)$ with various η's across the iterations of HQ. In Table 2,

$$\varrho = \frac{R_1(\boldsymbol{w}^1, b^1) - R_1(\boldsymbol{w}^{10}, b^{10})}{R_1(\boldsymbol{w}^1, b^1) - R_1(\boldsymbol{w}^{30}, b^{30})} \times 100\%,$$

which is used to show the decrease percentage of $R_1(\boldsymbol{w}^{10}, b^{10})$ among the first 30 iterations. First, we can see from Table 2 that HQ converges or nearly converges within 30 iterations in all the cases, which, in turn, confirms that Proposition 2 holds. Second, according to ϱ, we show that $(\boldsymbol{w}^{10}, b^{10})$ can be viewed as a good approximation to the solution of (5). We also do such experiments under other parameter settings (the results are in the supplementary material) and find that the above conclusions also establish in all the cases. By balancing the computational complexity and the convergence of HQ, we propose to set $S = 10$ in the following experiments and we will find that RSVM-RHHQ with $S = 10$ can give satisfactory performance. To summarize, HQ optimization method is effective in optimizing (5).

4.2. Comparison of weight functions

In this section, we intuitively compare the different weight functions discussed in Section 3.2. We generate an artificial dataset which is shown in Fig. 2. Squares are negatives and crosses are positives. Negatives are drawn from the Gaussian distribution $\mathcal{N}([-1;\ 0], [0.13,\ 0.06;\ 0.06,\ 0.26])$ and positives are drawn from $\mathcal{N}([1;\ 0], [0.13,\ 0.06;\ 0.06,\ 0.26])$. Negatives and positives have equal prior probabilities. We flip the labels of three samples and treat them as outliers. For clarity, we call WSVM using (17) *WSVM-* (17) for short and the same goes for the other two. We run SVM, RSVM-RHHQ, WSVM- (17), WSVM- (18) and WSVM- (19) on this artificial dataset. Linear kernel is used and the regularization parameters of the five methods are kept the same. For RSVM-RHHQ, $\eta = 2$, $S = 10$ and $\boldsymbol{v}^0 = -\mathbf{1}$; for WSVM- (17), $\Delta = 0.001$; and for WSVM- (18), $\eta = 0.7$. Notice that WSVM- (17), WSVM- (18) and WSVM- (19) only update the weights once.

First, the decision boundary of SVM is seriously biased from the Bayes decision boundary because of the outliers, which shows that SVM is not robust. Second, RSVM-RHHQ can effectively reduce the influence of the outliers while WSVM- (17), WSVM- (18) and WSVM- (19) fail. We learn from Fig. 2(a) that the weights of the outliers are much smaller than those of normal samples. However, in Figs. 2(b)–(d), the weights of the outliers are comparable with or even greater than those of normal samples. In this toy example, the assumption of (17)–(19), which is that outliers locate farther from the decision boundary of SVM than normal training samples, does not hold and thus, WSVM- (17), WSVM- (18) and WSVM- (19) become ineffective. To conclude, we believe that our weight function (16) is more reasonable than (17)–(19).

4.3. Comparison of robustness

In this section, we conduct experiments to compare the robustness of SVM, some existing robust SVMs and RSVM-RHHQ. First, we would like to list all the classifiers and clarify their parameter settings so as to make the comparison clear

- SVM [19].
- WSVM- (19) [8].
- Robust SVM based on the ramp loss (*RSVM-Ramp*) [5,12,14,16]. We apply the outlier-path algorithm[3] [16] to optimize RSVM-Ramp.
- RSVM-RHHQ: According to the discussion in Section 4.1, $S = 10$. η is in charge of the upper bound of $l_{rhinge}(z)$, and we compare the results of RSVM-RHHQs with different η's ($\eta = 0.2$, $\eta = 0.5$, $\eta = 1$, $\eta = 2$, $\eta = 3$ and cross validation on η) in order to know how to set η. Table 3 shows the loss bounds of $l_{rhinge}(z)$ with different η's and we can learn that when $\eta = 3$, the loss bound of $l_{rhinge}(z)$ is very close to 1, which is the loss bound of the 0–1 loss function [30] (see Fig. 1). $\boldsymbol{v_i}^0$ is initialized as follows [10]:

$$-v_i^0 = \frac{2}{1 + \exp(\theta\|\boldsymbol{x}_i - \bar{\boldsymbol{x}}\|_2)},$$

where $\theta > 0$ is a given constant and $\bar{\boldsymbol{x}}$ is the class center of the class that $\boldsymbol{x}_i$ belongs to.

We use 10-fold cross validation (*CV*), and at each run, we randomly choose 30% of training samples to form a validation dataset which is then used to tune the hyper-parameters of all the classifiers. In order to study the robustness of the different classifiers, we randomly introduce label noise into the training and validation datasets [4,5]. We consider 0%, 15% and 30% label noise levels. For example, 15% label noise means that we randomly flip 15% of the labels in the training and validation datasets respectively. Tables 4 and 5 show the classification results of all the classifiers with linear and the Gaussian kernels respectively. Fig. 3 presents the critical difference (*CD*) diagrams of the Friedman test with Nemenyi post-hoc test at the 5% significance level [31]. The Friedman test with Nemenyi post-hoc test is a nonparametric test which is commonly used to compare multiple classifiers over multiple datasets. In Fig. 3, the lower the rank is, the better the classifier is, and if the difference between the ranks of two classifiers is greater than CD, their performance is considered to be significantly different. In Fig. 3, we use a thick line to connect the classifiers that are not significantly different.

4.3.1. Discussion about η in $l_{rhinge}(z)$

We know from Fig. 1(b) and Table 3 that η controls the upper bound of $l_{rhinge}(z)$. In this section, we compare the classification results of RSVM-RHHQs with different η settings so as to draw a conclusion on how to choose η. According to Fig. 3, we know that except RSVM-RHHQ (CV on η), RSVM-RHHQ ($\eta = 0.2$) and RSVM-RHHQ ($\eta = 0.5$) generally perform better on the datasets without outliers, while RSVM-RHHQ ($\eta = 2$) and RSVM-RHHQ ($\eta = 3$) generally have better performance on the datasets with outliers. The superiority of RSVM-RHHQ ($\eta = 0.2$) and RSVM-RHHQ ($\eta = 0.5$) on the datasets without outliers is consistent with the fact that SVM also performs well when there are no outliers. However, when there are outliers, $l_{rhinge}(z)$ with $\eta = 2$ and $\eta = 3$ can place much less loss on outliers and thus, the decision boundaries of RSVM-RHHQ ($\eta = 2$) and RSVM-RHHQ ($\eta = 3$) are less influenced by outliers. We think that this is the reason that RSVM-RHHQ ($\eta = 2$) and RSVM-RHHQ ($\eta = 3$) show their advantage on the datasets with outliers. RSVM-RHHQ ($\eta = 2$) generally performs slightly better than RSVM-RHHQ ($\eta = 3$) on the datasets with outliers. To conclude, we suggest to choose small η (around 0.5) when there are no outliers and choose large η (around 2) when there are outliers in the datasets.

4.3.2. Comparison among classifiers

We learn from Tables 4 and 5 that when there are no outliers, the performance of all the classifiers is comparable to each other on most of the datasets. However, on the datasets with outliers, we know from Fig. 3 that WSVM- (19), RSVM-Ramp and RSVM-RHHQs generally perform better than SVM, which proves that the existing techniques are effective in improving the robustness of SVM. Besides, RSVM-RHHQ (CV on η) are significantly more robust than SVM in all the cases with outliers. WSVM- (19) can improve the robustness of SVM, but compared with our methods, its improvement is limited which, we think, is due to its heuristic weight function. RSVM-Ramp is a popular robust SVM and there are several optimization methods for it. Tables 4 and 5 indicate that the superiority of RSVM-Ramp is more evident on the datasets with 30% label noise. But its performance still does not surpass that of RSVM-RHHQs as Fig. 3 suggests. Furthermore, RSVM-Ramp with the Gaussian kernel renders poor results on the Musk (D6) and Protein (D9) datasets, which is possibly due to the more complex optimization procedure of $l_{ramp}(z)$.

4.3.3. Discussion about sparseness

Sparseness is an important property of SVM that is related with training time, testing time and generalization performance [1,32]. Table 6 shows the mean support vector ratios of all the classifiers over all the datasets. We learn from Table 6 that the sparseness of SVM is greatly affected when there are outliers in the datasets. Although WSVM- (19) can improve the robustness of SVM, it has little effect on improving the sparseness of SVM. However, both RSVM-Ramp and RSVM-RHHQs can improve the sparseness of SVM in the presence of outliers, which is partly because that they both place much less emphasis on outliers [5,12,14]. Generally, the larger η is, the better the sparseness of RSVM-RHHQ is.

According to the above experimental results, we come to the conclusion that RSVM-RHHQ not only improves the robustness of SVM but also has better sparseness than SVM.

5. Conclusion

In this paper, based on C-loss, a new loss function called the rescaled hinge loss is proposed, which is a monotonic, bounded and nonconvex loss that is robust to outliers. Furthermore, we derive that it becomes $l_{hinge}(z)$ as its scaling parameter η approaches 0. Then, we develop a new robust SVM called RSVM-RHHQ from the rescaled hinge loss by utilizing HQ optimization method. We find that RSVM-

[3] The C++ code: http://www-als.ics.nitech.ac.jp/code/index.php?FrontPage

RHHQ is equivalent to an iterative WSVM and in turn, we explain the robustness of iterative WSVM from a loss function perspective. According to the experimental results, RSVM-RHHQ is effective in dealing with outliers and can perform better than SVM and the existing robust SVM techniques on the datasets that have outliers. Moreover, the sparseness of RSVM-RHHQ is also better than that of SVM.

In fact, any classifier with an unbounded and convex loss may suffer from outliers. Although our focus of this paper is on SVM, the operation of rescaling can also be applied to other convex losses. In the future, we will explore the effect of rescaling on some other convex loss functions, such as the exponential loss and the logistic loss [30].

Acknowledgments

This work was supported by NSFC grant # 61273196 and China Scholarship Council.

Appendix A. Derivation of (7)

According to the definition of conjugate function [28], the conjugate function $g^*(u)$ of $g(v) = -v\log(-v) + v$, where $v < 0$, is

$$g^*(u) = \sup_{v<0}\{uv - g(v)\} = \sup_{v<0}\{uv + v\log(-v) - v\}. \tag{A.1}$$

Since $uv + v\log(-v) - v$ is a concave function with respect to v, we can easily obtain the solution of (A.1) by setting the derivative of $uv + v\log(-v) - v$ to 0, which is $v = -\exp(-u) < 0$. Then, we have that $(uv + v\log(-v) - v)|_{v=-\exp(-u)} = \exp(-u)$. Consequently, we derive that

$$g^*(u) = \sup_{v<0}\{uv + v\log(-v) - v\} = \exp(-u),$$

where the supremum is achieved at $v = -\exp(-u) < 0$. If we define $u = \eta l_{hinge}(z)$, we have that

$$g^*(\eta l_{hinge}(z)) = \sup_{v<0}\{\eta l_{hinge}(z)v + v\log(-v) - v\} = \exp(-\eta l_{hinge}(z)),$$

where the supremum is achieved at $v = -\exp(-\eta l_{hinge}(z)) < 0$.

Appendix B. Supplementary data

Supplementary data associated with this article can be found in the online version at http://dx.doi.org/10.1016/j.patcog.2016.09.045.

References

[1] V. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.
[2] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (September (3)) (1995) 273–297. http://dx.doi.org/10.1007/BF00994018.
[3] G. Wahba, Support vector machines, reproducing Kernel Hilbert spaces and the randomized gacv, in: Advances in Kernel Methods-Support Vector Learning, 1999, pp. 69–88.
[4] B. Frenay, M. Verleysen, Classification in the presence of label noise: a survey, IEEE Trans. Neural Netw. Learn. Syst. 25 (May (5)) (2014) 845–869. http://dx.doi.org/10.1109/TNNLS.2013.2292894.
[5] Y. Wu, Y. Liu, Robust truncated hinge loss support vector machines, J. Am. Stat. Assoc. 102 (September (479)) (2007) 974–983. http://dx.doi.org/10.1198/016214507000000617.
[6] T. Hastie, J. Friedman, R. Tibshirani, The Elements of Statistical Learning: Data Mining, Inference and Prediction, 2nd ed., Springer, New York, 2009.
[7] Q. Song, W. Hu, W. Xie, Robust support vector machine with bullet hole image classification, IEEE Trans. Syst Man Cybern. Part C: Appl. Rev. 32 (November (4)) (2002) 440–448. http://dx.doi.org/10.1109/TSMCC.2002.807277.
[8] Y. Wu, Y. Liu, Adaptively weighted large margin classifiers, J. Comput. Graph. Stat. 22 (May (2)) (2013) 416–432. http://dx.doi.org/10.1080/10618600.2012.680866.
[9] C.-F. Lin, S.-D. Wang, Fuzzy support vector machines, IEEE Trans. Neural Netw. 13 (March (2)) (2002) 464–471. http://dx.doi.org/10.1109/72.991432.
[10] R. Batuwita, V. Palade, Fsvm-cil: fuzzy support vector machines for class imbalance learning, IEEE Trans. Fuzzy Syst. 18 (June (3)) (2010) 558–571. http://dx.doi.org/10.1109/TFUZZ.2010.2042721.
[11] H. Ding, J. Xu, Random gradient descent tree: a combinatorial approach for svm with outliers, in: Proceedings of AAAI Conference on Artificial Intelligence (AAAI), 2015.
[12] R. Collobert, F. Sinz, J. Weston, L. Bottou, Trading convexity for scalability, in: Proc. of International Conference on Machine Learning (ICML), 2006, pp. 201–208. http://dx.doi.org/10.1145/1143844.1143870.
[13] L. Xu, K. Crammer, D. Schuurmans, Robust support vector machine training via convex outlier ablation, in: Proceedings of AAAI Conference on Artificial Intelligence (AAAI), 2006, pp. 536–542.
[14] S. Ertekin, L. Bottou, C.L. Giles, Nonconvex online support vector machines, IEEE Trans. Pattern Anal. Mach. Intell. 33 (February (2)) (2011) 368–381. http://dx.doi.org/10.1109/TPAMI.2010.109.
[15] X. Huang, L. Shi, J.A. Suykens, Ramp loss linear programming support vector machine, J. Mach. Learn. Res. 15 (January (1)) (2014) 2185–2211.
[16] S. Suzumura, K. Ogawa, M. Sugiyama, I. Takeuchi, Outlier path: A homotopy algorithm for robust svm, in: Proceedings of International Conference on Machine Learning (ICML), 2014, pp. 1098–1106.
[17] M. Nikolova, M.K. Ng, Analysis of half-quadratic minimization methods for signal and image recovery, SIAM J. Sci. Comput. 27 (March (3)) (2005) 937–966. http://dx.doi.org/10.1137/030600862.
[18] A. Singh, R. Pokharel, J.C. Principe, The c-loss function for pattern classification, Pattern Recognit. 47 (January (1)) (2014) 441–453. http://dx.doi.org/10.1016/j.patcog.2013.07.017.
[19] C.-C. Chang, C.-J. Lin, Libsvm: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (April (3)) (2011) 1–27. http://dx.doi.org/10.1145/1961189.1961199.
[20] W. Liu, P.P. Pokharel, J.C. Principe, Correntropy: properties and applications in non-gaussian signal processing, IEEE Trans. Signal Process. 55 (November (11)) (2007) 5286–5298. http://dx.doi.org/10.1109/TSP.2007.896065.
[21] J.C. Principe, Information Theoretic Learning: Renyi's Entropy and Kernel Perspectives, Springer, New York, 2010.
[22] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Trans. Pattern Anal. Mach. Intell. 31 (February (2)) (2009) 210–227. http://dx.doi.org/10.1109/TPAMI.2008.79.
[23] R. He, W.S. Zheng, B.-G. Hu, Maximum correntropy criterion for robust face recognition, IEEE Trans. Pattern Anal. Mach. Intell. 33 (August (8)) (2011) 1561–1576. http://dx.doi.org/10.1109/TPAMI.2010.220.
[24] R. He, W.-S. Zheng, B.-G. Hu, X.-W. Kong, A regularized correntropy framework for robust pattern recognition, Neural Comput. 23 (August (8)) (2011) 2074–2100.
[25] R. Pokharel, J.C. Principe, Kernel classifier with correntropy loss, in: Proceedings of International Joint Conference on Neural Networks (IJCNN), 2012, pp. 1–6.http://dx.doi.org/10.1109/IJCNN.2012.6252721.
[26] T.V. Gestel, J.A. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B.D. Moor, J. Vandewalle, Benchmarking least squares support vector machine classifiers, Mach. Learn. 54 (January (1)) (2004) 5–32. http://dx.doi.org/10.1023/B:MACH.0000008082.80494.e0.
[27] X.-T. Yuan, B.-G. Hu, Robust feature extraction via information theoretic learning, in: Proceedings of International Conference on Machine Learning (ICML), 2009, pp. 1193–1200. http://dx.doi.org/10.1145/1553374.1553526.
[28] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, Cambridge, 2004.
[29] M. Lichman, UCI Machine Learning Repository. URL ⟨http://archive.ics.uci.edu/ml⟩ (2013).
[30] T. Zhang, Statistical behavior and consistency of classification methods based on convex risk minimization, Ann. Stat. 32 (February (1)) (2004) 56–85.
[31] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (December) (2006) 1–30.
[32] S.-H. Yang, B.-G. Hu, A stagewise least square loss function for classification, in: Proceedings of the SIAM International Conference on Data Mining (SDM), 2008, pp. 120–131.

Guibiao Xu now is a Ph.D. candidate in the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, Beijing, China. Before that, he received the Bachelor degree from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2011. From September 2013 to September 2015, he was a visiting student in the Computational NeuroEngineering Laboratory (CNEL) at the University of Florida. His research interests are machine learning and data mining.

Zheng Cao is currently a Ph.D. student in the Computational NeuroEngineering Laboratory (CNEL) at the University of Florida. He received his Bachelor degree from Nanjing University of Science and Technology in 2010, and Master degree from University of Wyoming in 2012. His current research interests are machine learning and computer vision, with applications in marine animal detection and classification.

Bao-Gang Hu received the M.Sc. degree from the University of Science and Technology, Beijing, China, in 1983, and the Ph.D. degree from McMaster University, Hamilton, ON, Canada, in 1993, both in mechanical engineering. He is currently a Professor with the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, Beijing, China. From 2000 to 2005, he was the Chinese Director of Chinese-French Joint Laboratory for Computer Science, Control and Applied Mathematics (LIAMA). His current research interests include pattern recognition and plant growth modeling. He is a Senior Member of the IEEE.

Jose C. Principe is a Distinguished Professor of Electrical and Computer Engineering and Biomedical Engineering at the University of Florida. He is also the BellSouth

Professor and Founding Director of Computational NeuroEngineering Laboratory (CNEL), University of Florida. His primary research interests are advanced signal processing with information theoretic criteria (entropy and mutual information), adaptive models in the reproducing kernel Hilbert spaces (RKHS) and the application of these advanced algorithms in Brain Machine Interfaces (BMI). Dr. Principe is a Fellow of the IEEE, ABME and AIBME. He is the past Editor-in-Chief of the IEEE Transactions on Biomedical Engineering, past Chair of the Technical Committee on Neural Networks of the IEEE Signal Processing Society and past President of the International Neural Network Society. He received the IEEE EMBS Career Award, and the IEEE Neural Network Pioneer Award. He has more than 600 publications and 30 patents.