On Multicategory Truncated-Hinge-Loss Support Vector Machines

Yichao Wu and Yufeng Liu

Abstract. With its elegant margin theory and accurate classification performance, the Support Vector Machine (SVM) has been widely applied in both machine learning and statistics. Despite its success and popularity, it still has some drawbacks in certain situations. In particular, the SVM classifier can be very sensitive to outliers in the training sample. Moreover, the number of support vectors (SVs) can be very large in many applications. To solve these problems, [WL06] proposed a new SVM variant, the robust truncated-hinge-loss SVM (RSVM), which uses a truncated hinge loss. In this paper, we apply the operation of truncation on the multicategory hinge loss proposed by [LLW04]. We show that the proposed robust multicategory truncated-hinge-loss SVM (RMSVM) is more robust to outliers and deliver more accurate classifiers using a smaller set of SVs than the original multicategory SVM (MSVM) proposed by [LLW04].

1. Introduction

As a supervised learning technique, classification is an important tool for statistical data analysis. Among many classification methods, the Support Vector Machine (SVM) is a popular one and has enjoyed great success in many applications [Vap98, CST00]. The SVM was first invented by Vapnik and his colleagues using an elegant large margin theory. It is now known that the SVM can be fit in the regularization framework of Loss + Penalty using the hinge loss [Wah99]. In the regularization framework, the loss function is used to ensure fidelity of the resulting model to the data. The penalty term in regularization helps to avoid overfitting of the resulting model. A tuning parameter is typically used to balance these two components. Besides the SVM, many other classification methods belong to the regularization framework. For example, the penalized logistic regression [LWX+00, ZH05] and the AdaBoost [FHT00] use the logistic loss and the exponential loss respectively.

Despite its success, the SVM has been shown to have some drawbacks for difficult learning problems [WL06]. One drawback of the SVM classifier is that it tends to be sensitive to noisy training data. The reason is because SVM uses...
the unbounded hinge loss and consequently the resulting classifiers may be affected by points far away from their own classes, namely “outliers” in the training data. Another drawback of the SVM is that the number of SVs can be very large for many problems, especially for difficult classification problems or problems with a large number of input variables. To overcome these problems, [WL06] suggested to truncate the hinge loss and proposed the robust truncated-hinge-loss SVM (RSVM) based on the bounded truncated hinge loss. They showed that the RSVM is more robust to outliers using a smaller set of SVs than the original SVM.

In this paper, we focus on multicategory SVM (MSVM) and apply the operation of truncation on the multicategory hinge loss proposed by [LLW04]. We show that the proposed truncated multicategory hinge loss preserves Fisher consistency. Moreover, the proposed robust multicategory truncated-hinge-loss SVM (RMSVM) is more robust to outliers in the training data than the original MSVM. Furthermore, the RMSVM retains the SV interpretation and it often selects much fewer number of SVs than the MSVM.

Although truncation helps to robustify the MSVM, the associated optimization problem becomes nonconvex minimization. We propose to apply the d.c. algorithm to solve the nonconvex problem via a sequence of convex subproblems. Our numerical experience suggests that the d.c. algorithm works effectively.

The rest of the paper is organized as follows: In Section 2, we briefly review the SVM methodology and introduce the RMSVM. In Section 3, we develop a numerical algorithm for the RMSVM via the d.c. algorithm. We also give the SV interpretation of the RMSVM. In Section 4, we present numerical examples to demonstrate effectiveness of the truncated hinge loss. We conclude the paper with Section 5.

2. Multicategory Support Vector Machine and Its Robust Variant

2.1. Multicategory Support Vector Machine. For a $k$-class classification problem, we are given a training sample $\{(x_i, y_i) : i = 1, 2, \cdots, n\}$ which is distributed according to some unknown probability distribution function $P(x, y)$, with $p_j(x) = P(Y = j|X = x)$. Here $x_i \in \mathcal{S} \subset \mathbb{R}^d$ and $y_i; i = 1, \ldots, n$, denote the input vectors and output labels respectively, where $n$ is the sample size, and $d$ is the dimensionality of the input space. We label $y$ as $\{1, 2, \cdots, k\}$. Clearly, the Bayes rule is given by argmax$_{j=1,\ldots,k} p_j(x)$ which delivers the minimum expected misclassification rate.

Denote $f = (f_1, f_2, \cdots, f_k)$ to be the decision function vector, where each component represents one class and maps from $\mathcal{S}$ to $\mathbb{R}$. We use argmax$_{j=1,\ldots,k} f_j$ as the classifier which classifies a new input vector $x$ into the class with the largest $f_j(x)$. Let $f_j(x) = h_j(x) + b_j$ with $h_j \in H_K$, where $H_K$ is a reproducing kernel Hilbert space (RKHS) induced by the positive definite kernel $K(\cdot, \cdot)$.

To achieve multicategory classification, the standard MSVM proposed by [LLW04] solves the following optimization problem

\[
\min_f \frac{C}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} I(y_i \neq j) [1 + f_j(x_i)] + \frac{1}{2} \sum_{j=1}^{k} \|f_j\|^2, \tag{2.1}
\]

under the sum-to-zero constraint $\sum_{j=1}^{k} f_j(x) = 0$, where $C > 0$ is a tuning parameter. Note that nonstandard learning can be achieved by assigning different
misclassification costs. For simplicity, we will focus on standard learning in this article.

The critical aspect of the MSVM formulation in (2.1) is the multicategory hinge loss \( \sum_{j=1}^{k} I(y \neq j)[1 + f_j(x)]_+ \) subject to \( \sum_{j=1}^{k} f_j(x) = 0 \). It can be showed that the minimizer of \( E[\sum_{j=1}^{k} I(Y \neq j)[1 + f_j(X)]_+] \) subject to the sum-to-zero constraint is \( f^*(x) \) with \( f_j(x) = k - 1 \) if \( j = \arg \max_{j=1,\ldots,k} P_j(x) \) and \(-1\) otherwise [LLW04]. As a result, the MSVM formulation in (2.1) yields an extension of the binary SVM with Fisher consistency [c.f. Lin02].

Using the representer theorem [KW71, Wah99], \( f_j(x) \) can represented as \( b_j + \sum_{i'=1}^{n} K(x_i, x_{i'}) v_{i'j} \). Thus we have

\[
(2.2) \quad f_j(x_i) = b_j + \sum_{i'=1}^{n} K(x_i, x_{i'}) v_{i'j} = b_j + K^T_i v_j,
\]

where \( K_i = (K(x_i, x_1), K(x_i, x_2), \ldots, K(x_i, x_n))^T \) and \( v_j = (v_{1j}, \ldots, v_{nj})^T \). After plugging (2.2) into (2.1), the MSVM problem in (2.1) becomes

\[
(2.3) \quad \min_{\{v_j, b_j: j = 1, \ldots, k\}} \frac{C}{n} \sum_{i=1}^{n} \sum_{j \neq y_i} [b_j + K^T_i v_j + 1]_+ + \frac{1}{2} \sum_{j=1}^{k} v_j^T K v_j,
\]

subject to \( e \sum_{j=1}^{k} b_j + K \sum_{j=1}^{k} v_j = 0 \).

where \( K \) denotes the kernel matrix with the \((i, i')\) element being \( K(x_i, x_{i'}) \) and \( e = (1, 1, \ldots, 1)^T \) is a vector of length \( n \). Problem (2.3) can be solved using quadratic programming (QP) in a similar way as the binary SVM.

### 2.2. Robust Truncated-Hinge-Loss MSVM

Denote \( H_{-1}(u) = [1 + u]_+ \). Then the multicategory hinge loss in (2.1) can expressed as \( \sum_{j=1}^{k} I(y \neq j)[1 + f_j(x)]_+ = \sum_{j=1}^{k} I(y \neq j) H_{-1}(f_j(x)) \) subject to \( \sum_{j=1}^{k} f_j(x) = 0 \). For a given training pair \((x, y)\), this loss uses \( H_{-1}(f_j(x)) \) to encourage \( f_j(x) \) to be negative and thus \( f_j(x) \) to be positive by the sum-to-zero constraint. Notice that \( H_{-1}(u) \) increases linearly with \( u \) when \( u \geq -1 \). This implies that the loss will put emphasis on untypical points which are far away from their own classes, namely outliers. This is undesirable since the resulting classification boundary should not be greatly influenced by outliers.

To overcome the difficulty of outliers, we consider to decrease the impact of outliers by truncating the unbounded loss function. In particular, we consider to truncate \( H_{-1}(u) \). Denote \( H_s(u) = [u - s]_+ \) and define a truncated loss \( T_s(u) = H_{-1}(u) - H_s(u) \). Figure 1 displays the loss functions \( H_{-1}(u) \) and \( T_s(u) \). As we can see from the plot, \( T_s(u) \) becomes flat once \( u \geq s \). Consequently, \( T_s(f_j(x)) \) treats \( f_j(x) \) equally once it is greater than \( s \) and therefore it may yield more robust classifiers than the original function \( H_{-1} \). Using the truncated loss function \( T_s \), the new proposed RMSVM solves the following optimization problem

\[
(2.5) \quad \min_{f} \frac{C}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} I(y_i \neq j) T_s(f_j(x_i)) + \frac{1}{2} \sum_{j=1}^{k} ||f_j||^2,
\]

under the sum-to-zero constraint \( \sum_{j=1}^{k} f_j(x) = 0 \).
Theorem 2.1. The minimizer $f^*$ of $E[\sum_{j=1}^k I(Y \neq j)T_s(f_j(X))|X = x]$ subject to $\sum_{j=1}^k f_j(x) = 0$ satisfies that $\arg\max_j f_j^* = \arg\max_j p_j$, for any $s \geq 0$.

Proof. Without loss of generality, we assume $\max_j p_j > 1/k$ and $\arg\max_j p_j$ is unique, that is $\max_j p_j \neq \max_{j \neq i} p_j$. Denote $l_p = \arg\max_j p_j$.

Note that $E[\sum_{j=1}^k I(Y \neq j)T_s(f_j(X))|X = x] = \sum_{i=1}^k p_i(x) \sum_{j \neq i} T_s(f_j(x))$. Since $T_s$ is a non-decreasing function, we can conclude that $f_{l_p}^* \geq \max_{j \neq l_p} f_j^*$. Then it is sufficient to show that $f_{l_p}^* > 0$ and $f_j^* \leq 0$ for $j \neq l_p$. We will show it in two steps: (1). $f_{l_p}^* > 0$. (2). $f_j^* \leq 0$ for $j \neq l_p$.

To show part (1), it is equivalent to showing that $f_{l_p}^* = 0$ since $f_{l_p}^* = \max_j f_j^*$ and $\sum_{j=1}^k f_j = 0$. Suppose $f^* = 0$. Then $\sum_{i=1}^k p_i(x) \sum_{j \neq i} T_s(f_j^*(x)) = k - 1$. Consider another solution $f^0$ with $f_{l_p}^0 = k - 1$ and $f_j^0 = -1$ for $j \neq l_p$. Then $\sum_{i=1}^k p_i(x) \sum_{j \neq i} T_s(f_j^0(x)) = \sum_{j \neq l_p} p_j T_s(k - 1) = T_s(k - 1)(1 - p_{l_p}) < (k - 1)$. Thus $f_{l_p}^* > 0$.

For part (2), we show that $f^1$ with $f_{l_1}^1 > 0; l_1 \neq l_p$, cannot be the minimizer $f^*$. To this end, consider another solution $f^1$ with $f_{l_p}^1 = f_{l_p}^0 + f_{l_1}^1 + 1, f_{l_1}^1 = -1,$ and $f_j^1 = f_j^0, j \neq l_p, j \neq l_1$. Then $\sum_{i=1}^k p_i(x) \sum_{j \neq i} T_s(f_j^1(x)) - \sum_{i=1}^k p_i(x) \sum_{j \neq i} T_s(f_j^0(x)) = (1 - p_{l_p})[T(f_{l_p}) - T(f_{l_p})] - (1 - p_{l_1})[T(f_{l_1}) - T(f_{l_1})] = A$. The desired result follows from the fact that $A < 0$ as shown in the following four cases:

- $s \geq f_{l_p}^1$: $A = (1 - p_{l_p})(f_{l_1}^1 + 1) - (1 - p_{l_1})(f_{l_1}^1 + 1) < 0$.
- $f_{l_1}^1 \leq s < f_{l_p}^1$: $A < (1 - p_{l_p})(f_{l_1}^1 + 1) - (1 - p_{l_1})(f_{l_1}^1 + 1) < 0$.
- $f_{l_1}^1 \leq s < f_{l_p}^1$: $A = -(1 - p_{l_1})(f_{l_1}^1 + 1) < 0$.
- $0 \leq s < f_{l_1}^1$: $A = -(1 - p_{l_1})(s + 1) < 0$.

\[\square\]
Analogous to the MSVM formulation in (2.3), the proposed RMSVM is equivalent to solving the following minimization problem:

\[
\min_{\{v_j, b_j\}_{j=1}^k} \frac{C}{n} \sum_{i=1}^n \sum_{j \neq y_i} \left( [b_j + K_i^T v_{j} + 1]_+ - [b_j + K_i^T v_{j} - s]_+ \right) + \frac{1}{2} v_j^T K v_j
\]

subject to \( e \sum_{j=1}^k b_j + K \sum_{j=1}^k v_j = 0 \),

where \( s \geq 0 \) denotes the location of the truncation.

3. D.C. Algorithm

Since the truncated loss \( T_s \) is nonconvex, the optimization problem in (2.6) involves nonconvex minimization. Notice that \( T_s \) can be decomposed as the difference of two convex functions, \( H - \frac{1}{2} \) and \( H_s \). Using this property of the new loss function, we propose to apply the difference convex (d.c.) algorithm [AT97, LSD05] to solve the nonconvex optimization problem of the RMSVM. The d.c. algorithm solves the nonconvex minimization problem via minimizing a sequence of convex subproblems. As shown in [LSW05], the d.c. algorithm converges in finite steps and yields a local optimal solution of the original nonconvex minimization problem.

Next, we derive the d.c. algorithm for the proposed RMSVM and implement it via a sequence of QP. For simplicity of the notation, denote \( \Theta = \{v_j, b_j\}_{j=1}^k \). Then we break the objective function in (2.6) into two parts:

\[
\psi_{\text{vex}}(\Theta) = \frac{C}{n} \sum_{i=1}^n \sum_{j \neq y_i} \left( b_j + K_i^T v_{j} + 1 \right)_+ + \frac{1}{2} v_j^T K v_j
\]

\[
\psi_{\text{cav}}(\Theta) = -\frac{C}{n} \sum_{i=1}^n \sum_{j \neq y_i} \left( b_j + K_i^T v_{j} - s \right)_+.
\]

To apply the d.c. algorithm, we use a linear approximation on the concave part in the objective function. It is easy to see that

\[
\frac{\partial \psi_{\text{cav}}(\Theta)}{\partial v_j} = -\frac{C}{n} \sum_{i:y_i \neq j} \beta_{ij} K_i = -\frac{C}{n} \langle \beta_j \cdot L_j, v_j \rangle
\]

\[
\frac{\partial \psi_{\text{cav}}(\Theta)}{\partial b_j} = -\frac{C}{n} \sum_{i:y_i \neq j} \beta_{ij} = -\frac{C}{n} \langle \beta_j \cdot L_j, e \rangle,
\]

where \( L_j = (L_{1j}, L_{2j}, \ldots, L_{nj})^T \), \( \beta_j \cdot L_j \) denotes componentwise product, and

\[
\beta_{ij} = \begin{cases} 
1 & \text{if } b_j + K_i^T v_{j} - s > 0 \\
0 & \text{otherwise}.
\end{cases}
\]

Given the solution \( f^t \) at the \( t \)-th iteration, the objective at the \( (t + 1) \)-th iteration can be approximated by

\[
\frac{C}{n} \sum_{i=1}^n \sum_{j \neq y_i} \left( b_j + K_i^T v_{j} + 1 \right)_+ + \frac{1}{2} v_j^T K v_j + \sum_{j=1}^k \left( b_j \frac{\partial \psi_{\text{cav}}(\Theta^t)}{\partial b_j} + \left( \frac{\partial \psi_{\text{cav}}(\Theta^t)}{\partial v_j}, v_j \right) \right).
\]
Next we will convert (3.5) into a QP problem. To this end, we define $L$ to be a matrix with its $(i,j)$-th element $L_{ij} = I_{y_i \neq j}$, where $I_A$ is the indication function which takes value 1 if $A$ is true and 0 otherwise. Then at the $(t+1)$-th iteration, the d.c. algorithm of our RMSVM solves the following primal problem:

\[
\begin{align*}
(3.6) \quad & \min \frac{C}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} L_{ij} \xi_{ij} + \frac{1}{2} \sum_{j=1}^{k} v_j^T K v_j + \sum_{j=1}^{k} (b_j \frac{\partial \psi_{cav}(\Theta^T)}{\partial b_j} + \left( \frac{\partial \psi_{cav}(\Theta^T)}{\partial v_j}, v_j \right)) \\
(3.7) \quad & \text{s.t. } \xi_{ij} \geq 0 \\
(3.8) \quad & \xi_{ij} - (b_j + K^T_i v_j + 1) \geq 0 \\
(3.9) \quad & (\sum_{j=1}^{k} b_j) e + K (\sum_{j=1}^{k} v_j) = 0.
\end{align*}
\]

The corresponding Lagrangian function is

\[
L_D = \frac{C}{n} \sum_{i=1}^{n} \sum_{j \neq y_i} (C - \gamma_{ij} - \alpha_{ij}) \xi_{ij} + \sum_{j=1}^{k} b_j \left( \frac{\partial \psi_{cav}(\Theta^T)}{\partial b_j} + \sum_{i \neq j \neq y_i} \alpha_{ij} + \delta^T e \right) \\
+ \sum_{j=1}^{k} \left( \sum_{i \neq j \neq y_i} \alpha_{ij} K_{ij} + \frac{\partial \psi_{cav}(\Theta^T)}{\partial v_j} + K \delta, v_j \right) + \frac{1}{2} \sum_{i=1}^{n} \sum_{j \neq y_i} \alpha_{ij} + \frac{1}{2} \sum_{j=1}^{k} v_j^T K v_j,
\]

\[
= \frac{C}{n} \sum_{i=1}^{n} \sum_{j \neq y_i} (C - \gamma_{ij} - \alpha_{ij}) \xi_{ij} + \sum_{j=1}^{k} b_j \left( -\frac{C}{n} (\beta_j \cdot L_j)^T e + (\alpha_j \cdot L_j)^T e + \delta^T e \right) \\
+ \sum_{j=1}^{k} \left( K (\alpha_j \cdot L_j) - \frac{C}{n} K (\beta_j \cdot L_j) + K \delta, v_j \right) + \sum_{i=1}^{n} \sum_{j \neq y_i} \alpha_{ij} + \frac{1}{2} \sum_{j=1}^{k} v_j^T K v_j,
\]

where $\alpha_{ij} \geq 0$ and $\gamma_{ij} \geq 0$, $\delta = (\delta_1, \delta_2, \ldots, \delta_n)^T$ are Lagrangian coefficients.

Solving $\frac{\partial L_D}{\partial \xi_{ij}} = 0$, $\frac{\partial L_D}{\partial b_j} = 0$, and $\frac{\partial L_D}{\partial v_j} = 0$, we can get the corresponding dual QP problem. In particular, $\frac{\partial L_D}{\partial \xi_{ij}} = 0$ implies that $K v_j + K (\alpha_j \cdot L_j) + K \delta - \frac{C}{n} K (\beta_j \cdot L_j) = 0$. Due to the positive definitive kernel $K(\cdot, \cdot)$, this implies that

\[
(3.10) \quad v_j = -(\alpha_j \cdot L_j + \delta - \frac{C}{n} (\beta_j \cdot L_j)).
\]

Setting $\delta = -\frac{1}{k} \sum_{j=1}^{k} (\alpha_j \cdot L_j - \frac{C}{n} \beta_j \cdot L_j)$, $\alpha = \frac{1}{k} \sum_{j=1}^{k} (\alpha_j \cdot L_j)$, $\bar{\beta} = \frac{1}{k} \sum_{j=1}^{k} (\beta_j \cdot L_j)$, and $\delta = -(\bar{\alpha} - \frac{C}{n} \bar{\beta})$, we have

\[
(3.11) \quad v_j = -[\alpha_j \cdot L_j - (\bar{\alpha} - \frac{C}{n} \bar{\beta}) - \frac{C}{n} (\beta_j \cdot L_j)].
\]
After plugging (3.11) into $L_D$, we can rewrite the objective function of the corresponding dual problem as follows

$$
\sum_{j=1}^{k} \left\{ K(\alpha_j \cdot L_j) + K\delta - \frac{C}{n} K(\beta_j \cdot L_j), -(\alpha_j \cdot L_j + \delta - \frac{C}{n}(\beta_j \cdot L_j)) \right\}
+ \frac{1}{2} \sum_{j=1}^{k} \left\{ -(\alpha_j \cdot L_j + \delta - \frac{C}{n}(\beta_j \cdot L_j)), -K(\alpha_j \cdot L_j + \delta - \frac{C}{n}(\beta_j \cdot L_j)) \right\}
+ \sum_{i=1}^{n} \sum_{j \neq y_i} \alpha_{ij}
= \ - \frac{1}{2} \sum_{j=1}^{k} \left\{ (\alpha_j \cdot L_j + \delta - \frac{C}{n}(\beta_j \cdot L_j)), K(\alpha_j \cdot L_j + \delta - \frac{C}{n}(\beta_j \cdot L_j)) \right\}
+ \sum_{i=1}^{n} \sum_{j \neq y_i} \alpha_{ij} + \text{Constant}.
$$

Thus, the dual problem of the $(t+1)$-th iteration of our DCA for RMSVM is

$$(3.12) \min_{\Theta} \frac{1}{2} \sum_{j=1}^{k} \left[ (\alpha_j \cdot L_j - (\bar{\alpha} - \frac{C}{n}\bar{\beta}) - \frac{C}{n}(\beta_j \cdot L_j)), K(\alpha_j \cdot L_j - (\bar{\alpha} - \frac{C}{n}\bar{\beta}) - \frac{C}{n}(\beta_j \cdot L_j)) \right] - \sum_{i=1}^{n} \sum_{j \neq y_i} \alpha_{ij}$$

s.t.  

$$0 \leq \alpha_{ij} \leq \frac{C}{n}, i = 1, 2, \ldots, n, j \neq y_i$$

$$-\frac{C}{n}(\beta_j \cdot L_j)^{T} e + (\alpha_j \cdot L_j)^{T} e - (\bar{\alpha} - \frac{C}{n}\bar{\beta})^{T} e = 0, j = 1, 2, \ldots, k.$$  

Once the solution $\alpha$ of problem (3.12) is derived, the coefficients $v_j$ can be recovered using the following equation:

$$(3.15) v_j = -[\alpha_j \cdot L_j - (\bar{\alpha} - \frac{C}{n}\bar{\beta}) - \frac{C}{n}(\beta_j \cdot L_j)].$$

We are now left to solve $b_j$’s using linear programming (LP) once $v_j$’s are obtained at the $(t+1)$-th iteration. Denote $\tilde{f}_j(x_i) = K_j^{T} v_j$. Then $b_j$’s can be obtained by solving the following LP problem:

$$\min_{b} \frac{C}{n} \sum_{i=1}^{n} \sum_{j \neq y_i} [b_j + \tilde{f}_j(x_i) + 1]_+ + \sum_{j=1}^{k} b_j \frac{\partial \psi_{\text{cav}}(\Theta^t)}{\partial b_j}$$

subject to  

$$\sum_{j=1}^{k} b_j = 0.$$
Figure 2. For one typical training sample in Section 4, the left panel plots all observations with the black straight lines shown as the Bayes boundary. The middle and right panels display the classification boundaries obtained by using loss functions $H_{-1}$ and $T_1$, respectively, with SVs shown in red.

More explicitly,

$$
\begin{align*}
\min_b \quad & \frac{C}{n} \sum_{i=1}^n \sum_{j \neq y_i} \xi_{ij} + \sum_{j=1}^k b_j \frac{\partial \psi_{cav}(\Theta^t)}{\partial b_j} \\
\text{subject to} \quad & \sum_{j=1}^k b_j = 0 \\
& \xi_{ij} \geq 0, i = 1, 2, \ldots, n; j \neq y_i \\
& \xi_{ij} \geq b_j + \tilde{f}_j(x_i) + 1, i = 1, 2, \ldots, n; j \neq y_i.
\end{align*}
$$

We stop the algorithm when the objective function value in (2.6) converges.

4. Numerical Examples

Three-class nonlinear examples with $p = 2$ are generated in the following way:

First, generate $(x_1, x_2)$ uniformly over the unit disc $\{(x_1, x_2) : x_1^2 + x_2^2 \leq 1\}$. Define $\vartheta$ to be the radian phase angle measured counterclockwise from the ray from $(0, 0)$ to $(1, 0)$ to another ray from $(0, 0)$ to $(x_1, x_2)$. For a 3-class example, the class label $y$ is assigned as follows: $y = 1$ if $\left\lfloor \frac{k \vartheta}{2\pi} \right\rfloor + 1 = 1$ or 4; $y = 2$ if $\left\lfloor \frac{k \vartheta}{2\pi} \right\rfloor + 1 = 2$ or 3; $y = 3$ if $\left\lfloor \frac{k \vartheta}{2\pi} \right\rfloor + 1 = 5$ or 6. Next, randomly contaminate the data by randomly selecting $perc\% = 10\%$ or $20\%$ instances and changing their label indices to one of the remaining two classes with equal probabilities. This example was also considered in [WL06].

We apply the Gaussian kernel $K(x_1, x_2) = \exp(-\frac{(x_1, x_2)}{2\sigma^2})$. Here, two parameters need to be selected. The first parameter $C$ is chosen using a grid search. The second parameter $\sigma$ for the kernel is tuned among the first quartile, median, and the third quartile of the between-class pairwise Euclidean distances of training inputs ([BGL+00]).
Table 1. Results of the nonlinear examples in Section 4

<table>
<thead>
<tr>
<th>perc%</th>
<th>Loss</th>
<th>Test Error</th>
<th>#SV</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>$H_{-1}$</td>
<td>0.1712 (0.0198)</td>
<td>62.14 (13.43)</td>
</tr>
<tr>
<td></td>
<td>$T_1$</td>
<td>0.1595 (0.0203)</td>
<td>38.72 (16.05)</td>
</tr>
<tr>
<td>20%</td>
<td>$H_{-1}$</td>
<td>0.2793 (0.0270)</td>
<td>75.10 (10.60)</td>
</tr>
<tr>
<td></td>
<td>$T_1$</td>
<td>0.2672 (0.0251)</td>
<td>36.96 (12.49)</td>
</tr>
</tbody>
</table>

We have applied MSVMs with the unbounded loss $H_{-1}$ and the truncated loss $T_1$ for different contamination percentages (10% and 20%). Results based on 50 repetitions are reported in Table 1. From the table, we can see that RMSVMs give smaller testing errors while using fewer SVs than the MSVM. To visualize decision boundaries and SVs of the original MSVM and RMSVM, we choose one typical training sample and plot the results in Figure 2. The left panel shows the observations as well as the Bayes boundary. In the remaining two panels, boundaries using nonlinear learning with loss functions $H_{-1}$ and $T_1$ are plotted and their corresponding SVs are labelled in red. From the plots, we can see that the RMSVM uses much fewer SVs and at the same time yields more accurate classification boundaries than the MSVM.

5. Discussion

In this paper, we propose a robust version of MSVM, namely the RMSVM. The RMSVM uses the truncated hinge loss and delivers more robust classifiers than the MSVM. Our algorithm and numerical results show that the RMSVM has the interpretation of SVs and it tends to use a smaller yet more stable set of SVs than that of the MSVM.

References


[LLW04] Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines, theory, and application to the classification of microarray data and


