Speeding up SVM Decision Based on Mirror Points

Jiun-Hung Chen and Chu-Song Chen
Institute of Information Science, Academia Sinica, Taipei, Taiwan, R.O.C.
{jhchen,song}@iis.sinica.edu.tw

Abstract

In this paper, we propose a new method to speed up SVM decision based on the idea of mirror points. Decisions based on multiple simple classifiers, which are formed as a result of mirror pairs, are combined to approximate a single SVM. A dynamic programming-based method is used to find a suitable combination. Experimental results show that this method can increase classification efficiencies of SVM with comparable classification performances.

1 Introduction

The classification time of support vector machines (SVMs) [2][6] is proportional to the number of support vectors. Many works [1][3][5] were proposed to address on reducing its classification time by finding a simplified classifier where the number of vectors used in determining classification result is smaller than the number of support vectors. More specifically, they all focused on solving the reduced set problem [1] that will be introduced in Section 2.2.

Unlike previous works, our idea is to use several simple classifiers -"simple" in the sense that the number of vectors involved in determining a classification result is small, and decisions of these classifiers are combined to approximate an SVM. From the viewpoint of the number of parameters needed to be optimized, using simple classifiers has the advantage that the computation time for finding each simple classifier can be much less than that for solving the reduced set problem. In this paper, a kind of simple classifier is devised based the idea of mirror points. Particularly, only two vectors are required in determining a classification result for a simple classifier based on a pair of mirror points. In addition, such classifiers can be constructed from training data systematically. As for how to combine a set of chosen classifiers, it is a basically combinatorial problem and is intractable if a brute force method is used. In this paper, it is transformed to be an optimal-path finding problem in a specially designed graph, and a dynamic programming (DP)-based approach is used for it to find a good sub-optimal solution.

2 Review

Consider a two-class classification problem. Let $\Omega = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n) | x_i \in R^d, y_i \in \{-1, 1\}, i = 1, 2, ..., n\}$ be a set of input-output training data pairs where R^d space is referred to as the input space and is denoted by Λ in the following. The SVM [2][6] first projects the input vectors onto the feature space F by some nonlinear function $\phi : \Lambda \rightarrow F$ and then finds a linear separating hyperplane $H_{\tilde{w},b} : \tilde{w}^T \tilde{x} + b = 0$ in the feature space where $\tilde{x}, \tilde{w} \in F$ and $b \in R$. In SVM, $\tilde{w}$ and b are found by solving a quadratic programming problem, and $\tilde{w}$ has the following form

$$\tilde{w} = \sum_{i=1}^{n} \alpha_i y_i \phi(x_i), \tag{1}$$

where $\alpha_i \geq 0, i = 1, \ldots, n$. Notice that a training vector x_i with nonzero α_i is called a support vector and typically the number of support vectors is smaller than that of training examples. The classification rule of the SVM is

$$\begin{aligned} f_{H_{\tilde{w},b}}(x) &= sgn(\tilde{w}^T \phi(x_i) + b) \\ &= sgn(\sum_{i=1}^{n} \alpha_i y_i K(x, x_i) + b), \end{aligned} \tag{2}$$

where $K(x, x_i) = \phi(x)^T \phi(x_i)$. Notice that $K(.,.)$ is referred to as a Mercer's kernel [6] and commonly used Mercer's kernels include Gaussian RBF, polynomial, and sigmoidal functions. In the following, we will represent $f_{H_{\tilde{w},b}(x)}$ as $f(x)$ for brevity for cases without incurring confusion.

From (2), it can be observed the classification time is proportional to the number of support vectors. Hence, to speedup the SVM decision, many research works [1][3][5] focus on solving the *reduced set (RS) problem* [1], which is to find Υ^* defined below.

$$\Upsilon^* = arg \min_{\Upsilon} d(\Upsilon, \tilde{w}), \tag{3}$$

where

$$\Upsilon = \sum_{i=1}^{l} \beta_i \phi(\hat{x}_i), \tag{4}$$

1051-4651/02 $17.00 (c) 2002 IEEE

$d(\tilde{x}, \tilde{y}) = ||\tilde{x} - \tilde{y}||$ is the Euclidean distance between two vectors $\tilde{x}, \tilde{y}$ in the feature space F, and l is smaller than the number of support vectors. Then a set of $\beta_i \in R, \hat{x}_i \in \Lambda, i = 1, \ldots, l$ can be found by optimizing (3) into which (4) is incorporated. The number of parameters in this optimization problem is $l * (d + 1)$. While l or d is large, the computation time for solving the optimization problem is very demanding.

Some iterative methods including the reduced set method [1][5] and the regression method [3] have been proposed for solving the RS problem.

3 The Proposed Method

Assume that a linear separating hyperplane $H_{\tilde{w},b}$ in the feature space F is used as a classifier in the following discussions with $\tilde{w} = \sum_{i=1}^{n} \alpha_i y_i \phi(x_i)$ and b being obtained in advance. Given $v \in \Lambda$, the distance from its image $\phi(v)$ to a hyperplane $H_{\tilde{w},b}$, denoted by $d(\phi(v), H_{\tilde{w},b})$, is

$$d(\phi(v), H_{\tilde{w},b}) = \frac{|\tilde{w}^T \phi(v) + b|}{||\tilde{w}||} \tag{5}$$

Furthermore, its mirror vector, $\tilde{m_v}$, in the feature space F with respect to $H_{\tilde{w},b}$ is defined as follows.

$$\tilde{m_v} = \phi(v) - 2f(v)d(\phi(v), H_{\tilde{w},b})\frac{\tilde{w}}{||\tilde{w}||} \tag{6}$$

Given a pair of mirror points $(\phi(v), \tilde{m_v})$, define a classification rule $g_{\phi(v),\tilde{m_v}}(z)$ as follows

$$g_{\phi(v),\tilde{m_v}}(z) = \begin{cases} f(v) & \text{if } d(\phi(z), \phi(v)) \le d(\phi(z), \tilde{m_v}), \\ -f(v) & \text{otherwise,}. \end{cases} \tag{7}$$

where $z \in \Lambda$. Then the following property holds.

Property 1: $g_{\phi(v),\tilde{m_v}}(z) = f(z)$ for all $z \in \Lambda$.

From Property 1, it is known that classification results $f(z)$ and $g_{\phi(v),\tilde{m_v}}(z)$ are the same for all $z \in \Lambda$. This concept is shown in Figure 1. If the pre-image of $\tilde{m_v}$ can be clearly identified, then a single pair of mirror points, $(\phi(v), \tilde{m_v})$, can be used to construct an equivalent classifier of the SVM. However, the pre-image of $\tilde{m_v}$ may either not exist or require a complex representation. In this subsection, let us consider the case where $\tilde{m_v}$ is approximated by $\tilde{u}$. $\tilde{u}$ is referred to as the approximate mirror point of v. An approximate classification rule $AM^{\tilde{u}}_{\phi(v)}(z)$, which results in a linear classifier in the feature space, is defined as follows.

$$AM^{\tilde{u}}_{\phi(v)}(z) = \begin{cases} f(v) & \text{if } d(\phi(z), \phi(v)) \le d(\phi(z), \tilde{u}), \\ -f(v) & \text{otherwise.} \end{cases} \tag{8}$$

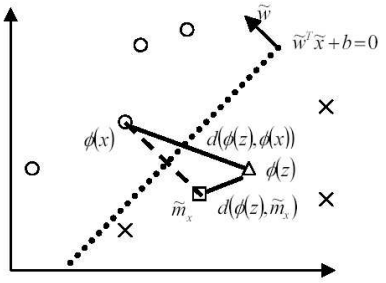


Figure 1. Linear classifiers based on mirror points. The dotted line $\tilde{w}^T\tilde{x} + b = 0$ is a linear separating hyperplane and the arrow points out to the region in which data are classified with positive labels. "O"s and "X"s are some data points that belong to the positive and negative classes, respectively. The mirror point of $\phi(x)$ is $\tilde{m}_x$. Then, z can be classified according to the distances $d(\phi(z), \phi(x))$ and $d(\phi(z), \tilde{m_x})$. If $d(\phi(z), \phi(x))$ is not larger than $d(\phi(z), \tilde{m_x})$, z is classified to the same class of point x. Otherwise, z is classified to the other class.

$AM^{\tilde{u}}_{\phi(v)}$ uses a single pair of mirror points to approximate f. In the following, we design a procedure **IAM** based on an integration of L AM-type classifiers (i.e., $AM^{\tilde{u_j}}_{\phi(v_j)}(z), j = 1, \ldots, L$) to approximate f in a more accurate way, and its pseudo codes are listed below.

Procedure IAM(**input**: z, **output**: result)
0. v_j and $\tilde{u}_j$ are given for all $j = 1, \ldots, L$.
10. result $\leftarrow 0$.
20. **For** (j=1 to L)
30. Compute $d(\phi(z), \tilde{u_j})$ and $d(\phi(z), \phi(v_j))$.
40. result$\leftarrow$result$+\Xi^{\tilde{u_j}}_{\phi(v_j)}(z)$
50. **END FOR**
60. result $\leftarrow sgn$(result),

where $\Xi^{\tilde{u}}_{\phi(v)}(z)$ is defined as

$$\Xi^{\tilde{u}}_{\phi(v)}(z) = \begin{cases} f(v)D^z_{\tilde{u},v} & \text{if } d(\phi(z), \phi(v)) \le d(\phi(z), \tilde{u}), \\ -f(v)D^z_{\tilde{u},v} & \text{otherwise.} \end{cases} \tag{9}$$

where $D^z_{\tilde{u},v} = |d(\phi(z), \tilde{u}) - d(\phi(z), \phi(v))|$.

Compared to other fast methods [1][3][5] for SVM decision, one distinct feature of our approximate version IAM is that, instead of finding a single $\tilde{u}$ for approximating $\tilde{m_v}$, decision combining of multiple simple classifiers based on mirror pairs (for example, (8) and (9)) is used for approximating the decision obtained from the SVM.

Two problems remained to be solved in the proposed methods are how to find the approximate pre-image of a mirror point and how to find a suitable combination of classifiers. These two problems are addressed in the next two subsections, respectively.

1051-4651/02 $17.00 (c) 2002 IEEE

3.1 Pre-image of a mirror point

Given $\tilde{m_v}$, the problem of finding (β^*, x^*) satisfying the following equation is considered:

$$(\beta^*, x^*) = arg \min_{\beta \in R, x \in \Omega_I} d(\beta\phi(x), \tilde{m_v}), \quad (10)$$

where $\Omega_I = \{x_1, \ldots, x_n\}$ is the input training data set. $\tilde{m_v}$ can then be approximated as $\tilde{u} = \beta^*\phi(x^*)$. We find (β^*, x^*) by investigating the training data set. First, for each x_i, β_i^* defined below needs to be found.

$$\beta_i^* = arg \min_{\beta_i \in R} d(\beta_i\phi(x_i), \tilde{m_v}) \quad (11)$$

In fact, analytic solutions of β_i^* can be obtained as shown below.

$$\begin{aligned} \beta_i^* =& \frac{R_1}{R_2} \\ d(\beta_i^*\phi(x_i), \tilde{m_v}) =& (R_3 - \frac{R_2^2}{R_1})^{\frac{1}{2}} \end{aligned} \quad (12)$$

where $R_1 = \phi(x_i)^T\phi(x_i)$, $R_2 = \tilde{m_v}^T\phi(x_i)$ and $R_3 = \tilde{m_v}^T\tilde{m_v}$. After finding β_i^* for each x_i contained in the training set, (β_i^*, x_i) with the minimal $d(\beta_i^*\phi(x_i), \tilde{m_v})$ among all $i = 1, \ldots, n$ is set to (β^*, x^*).

More generally, we can consider that

$$(\beta^{**}, x^{**}) = arg \min_{\beta \in R, x \in \Lambda} d(\beta\phi(x), \tilde{m_v}), \quad (13)$$

where $\tilde{u} = \beta^{**}\phi(x^{**})$, and an iterative method developed in [5] can be used for solving this approximate pre-image problem.

3.2 Combination of classifiers

It can be easily verified that the following classification rule is used to classify a given z in the procedure **IAM**.

$$f_{IAM}(z) = sgn(\sum_{i=1}^{L} \Xi_{\phi(v_i)}^{\tilde{u}_i}(z)). \quad (14)$$

How to find a suitable set $\{(v_1, \tilde{u}_1), (v_2, \tilde{u}_2), \ldots, (v_L, \tilde{u}_L)\}$ that best approximates f is addressed below.

Assume that K AM-type classifiers are given and whose distance-weighted functions are $\Xi_1(.), \ldots, \Xi_K(.)$, respectively. The problem is formulated as finding a suitable subset containing L classifiers, among the K classifiers ($L < K$), which has the best classification performance. The initial K classifiers can be generated by randomly choosing $v \in \Lambda$ and finding its $\tilde{u}$ that best approximates $\tilde{m_v}$ as described in Section 3.3, while in our work, they are generated by choosing from the input training data set. Let P be a mapping from $\{1, \ldots, L\}$ to $\{1, \ldots, K\}$. Then, $\Xi_{P(1)}, \Xi_{P(2)}, \ldots, \Xi_{P(L)}$ correspond to a set of L classifiers chosen from $\Psi = \{\Xi_1, \ldots, \Xi_K\}$, and to choose a set of L classifiers among the initial K ones is equivalent to find a P. Consider that it is expected that $sgn(\sum_{i=1}^{L} \Xi_{P(i)}(x_j)) = y_j$ for all $j = 1, \ldots, n$, in the ideal case. A performance index is then defined as

$$g(P) = \sum_{j=1}^{n} y_j(sgn(\sum_{i=1}^{L} \Xi_{P(i)}(x_j))), \quad (15)$$

and it is desired to find a P maximizing the performance index:

$$P^* = arg \max_P g(P) \quad (16)$$

To find the global optimal P^* is a difficult task, and a suboptimal $\hat{P}$ is found instead in our work, as shown in the following.

At first, we construct a graph G with L levels. For each level, there are K nodes. For each pair of adjacent levels $(l, l+1), l = 1, \ldots, L-1$, there is an edge linking each node on level l to each node on level $l+1$. Node $N_{i,l}$ denotes the node at level l and represents classifier Ξ_i. Initially, let $S_n(N_{i,1}) = [\Xi_i(x_1), \ldots, \Xi_i(x_n)]$ and $\Delta(N_{i,1}) = i, i = 1, \ldots, K$.

Then the following procedure is performed from $l = 1$ to $l = L-1$ to find $\hat{P}$. Given l, the following two steps are performed for each node $N_{m,l+1}, m = 1, \ldots, K$. First,

$$\Delta(N_{m,l+1}) = arg \max_i \sum_{j=1}^{n} y_j(sgn(S_n(N_{i,l})_j + \Xi_m(x_j))), \quad (17)$$

where $S_n(N_i, l)_j$ is the j-th component of $S_n(N_i, l)$. Then, we set $S_n(N_{i,l+1})$ by

$$S(N_{m,l+1}) = S(\Theta(N_{m,l+1})) + [\Xi_m(x_1), \ldots, \Xi_m(x_n)] \quad (18)$$

where $\Theta(N_{m,l+1}) = N_{\Delta(N_{m,l+1}),l}$.

After performing the above procedures iteratively, $\hat{P}$ is set as $\hat{P}(j) = Index(\Theta^{L-j+1}(N_{j^*,L})), j = 1, \ldots, L$, where $\Theta^t = \Theta \circ \Theta \circ \ldots \Theta$ is the composition of Θ t times, $Index(N_{i,l}) = i$, and $j^* = arg \max_j \sum_{i=1}^{n} y_i sgn(S(N_{j,L})_i)$. The associated classifiers are served for IAM where $\Xi_{\phi(v_j)}^{\tilde{u}_j} = \Xi_{\hat{P}(j)}$. Note that the above method can obtain the global optimal if $sgn(.)$ is modified to as an identical function in (15), (17) and the definition of j^*. In fact, in this case, the above method finds a path with the largest additive score based on DP if the score of a node $N_{i,l}$ is defined as $\sum_{j=1}^{n} \Xi_i(x_j)$. This DP-based method can not ensure to find P^* since $sgn(.)$ is not additive. However, it can help find a good suboptimal solution according to our experience.

4 Experimental Results

Two experimental results, where the first uses the Ripley data set [1] that is a synthetic example and the second uses a real ionosphere data set [2], are presented in this section. We use LIBSVM [3] for generating SVM classifiers. The approximate mirror points are all found off-line. The RBF kernels are adopted and the parameters γ and C used for training SVM for Ripley and ionosphere data sets are $(1, 100)$ and $(0.45, 10)$, respectively.

The correct rates for training and testing data of the Ripley data set using our method are 88% and 89%, respectively, which are comparable to the classification results 89.6% and 89.7% obtained by SVM [4]. From these experimental results, our method can achieve 19.5 times faster than SVM did with comparable results.

For ionosphere data set, we use ten-fold cross-validation method to verify the classification and speedup performance. The correct rates of SVM for the training and testing data are 100% and 94.6%, respectively. Fig. 2 shows the experimental results. For example, when the number of mirror vectors in use is 9.6, the speedup ratio is 17.9. The obtained correct ratios for training and testing data are 89.2% and 90.0%, respectively. In addition, the best correct rates for training and testing data are 93.7% and 93.7%, respectively, where the number of vectors involved in determining classifications is 27 (i.e the speedup ratio is 6.36).

5 Conclusion

We propose a new method to speed up the SVM based on the idea of mirror points. Compared to those in [1][3][5], a distinct feature of our method is that combination of decisions of multiple simple classifiers is used for approximating the decision of the SVM. Furthermore, our method can also be applied for RBF networks or classifiers whose normal vectors of their separating hyperplanes can be expressed like (1). One of our future work is to investigate that how to find better approximate mirror points.

Acknowledgments

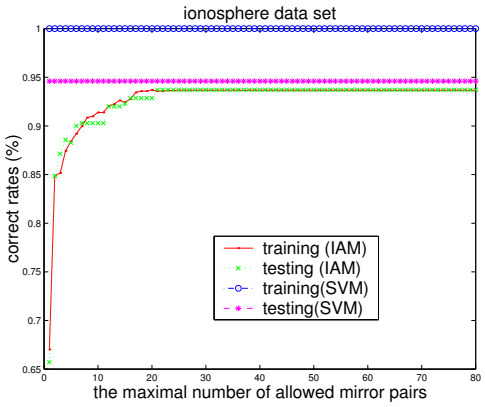
This work is supported in part by the National Science Council of Taiwan under Grant NSC 90-2213-E-001-033.

[1] Available at ftp://markov.stats.ox.ac.uk/pub/neural/papers.

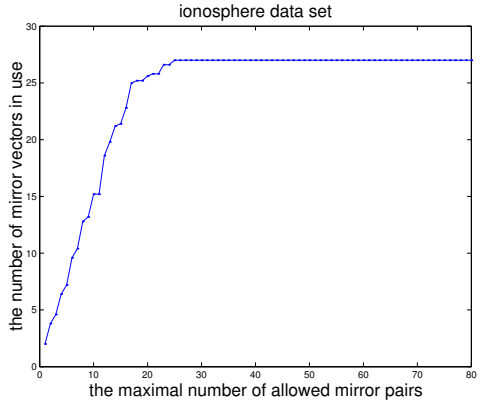
[2] Available at UCI Repository of machine learning databases http://www.ics.uci.edu/∼mlearn/MLRepository.html.

[3] Available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

[4] Our experimental setting on Ripley data set follows [3]. From their experimental results, 14 vectors and 33 vectors (i.e., about 5.5 and 2.33 times of speedup) were, respectively, used to approximate $\tilde{w}$ in order to meet their different criterions. However, they did not report classification results.



(a)



(b)

Figure 2. Speedup performance of ionosphere data set. (a)Correct rates vs. the maximal number of allowed mirror pairs. (b)The number of mirror vectors in use vs. the maximal number of allowed mirror pairs.

References

[1] C. Burges, "Simplified support vector decision rules," in *ICML'96*, pp. 71-77, San Mateo, CA, 1996.

[2] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Network*, vol. 12, pp. 181-201, 2001.

[3] E. Osuna and F. Girosi, "Reducing the run-time complexity of support vector machines," in B. Sch"olkopf, C.Burges and A. Smola (ed.), *Advances in Kernel Methods: Support Vector Learning*, MIT press, Cambridge, MA, p. 271-284, 1999.

[4] S. Romdhani, P. Torr, B. Schölkopf, and A. Blake, "Computationally Efficient Face Detection," in *ICCV'2001*, pp. 695-700, Vancouver, Canada.

[5] B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G.Rätsch, and A. J. Smola, Input space versus feature space in kernel-based methods, *IEEE Trans. Neural Networks*, vol. 10, pp. 1000-1017, 1999.

[6] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995.