A Fast Learning Algorithm for One-Class Support Vector Machine

Jia Jiong , Zhang Hao-ran
Institute of Computer Science,Zhejiang Normal University,Jinhua, China 321004
E-MAIL: jia@zjnu.cn

Abstract

Support vector machine (SVM) is a powerful tool to solve classification problems, this paper proposes a fast Sequential Minimal Optimization (SMO) algorithm for training one-class support vector regression (OCSVM), firstly gives a analytical solution to the size two quadratic programming (QP) problem, then proposes a new heuristic method to select the working set which leads to algorithm's faster convergence. The simulation results indicate that the proposed SMO algorithm can reduce the training time of OCSVM, and the performance of proposed SMO algorithm is better than that of original SMO algorithm.

1. Introduction

Support Vector Machines (SVM) have been shown to be highly effective in classification problem and other important learning tasks [1]. They are maximal-margin classifiers, rather than probabilistic as is Naïve Bayes. In the two-class formulation, the basic idea is to map feature vectors to a high dimensional space and to compute a hyperplane that not only separates the training vectors from different classes, but also maximizes this separation by making the margin as large as possible.

The standard SVM algorithm is a supervised learning algorithm. It requires labeled training data to create its classification rule. Scholkopf [2] proposes a method to adapt the SVM algorithm for one-class SVM(OCSVM), which only uses examples from one-class, instead of multiple classes, for training. This one-class SVM does not require its training set to be labeled to determine a decision surface. Whereas standard SVM tries to maximally separate two classes of data in feature space by a hyperplane, the one-class instead attempts to separate the entire set of training data from the origin, that is, it treats the origin as the only example from other classes. This is done by solving a quadratic program that penalizes any points not separated from the origin while simultaneously trying to maximize the distance of this hyperplane from the origin.

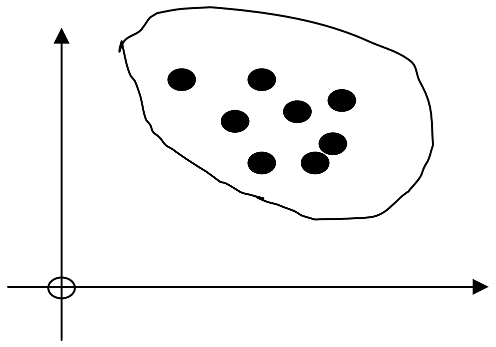
Figure 1. One-class SVM classifier, the origin is the only original member of the second class

The algorithm of one-class SVM is similar to the standard SVM algorithm in that it uses kernel functions to perform implicit mappings and dot products. It also uses the same kind of hyperplane for the decision surface. The solution is only dependent on the support vectors as well. The decision surface that is chosen is determined by solving an optimization problem that determines the "best" hyperplane under a set of criteria[3]. Stolfo and Wang[4] successfully apply the one-class SVM to masquerade detection and compare it with several of the other one-class techniques. Chen uses the one-class SVM for image retrieval[5].

Training a support vector machine(not only standard SVM but also one-class SVM) requires the solution of a very large quadratic programming (QP) optimization problem. Conventional QP methods, such as quasi-newton, primal-dual interior, are used to train SVM. For small datasets this is practical, but for larger datasets the QP method is impractical due to its high memory requirement and slow computation speed. Training a SVM on a large training set has become a bottleneck. Alternative techniques must be adopted to tackle this problem. John C. Platt[6] proposed a new algorithm for training classification support vector machines: Sequential Minimal Optimization, or SMO. SMO breaks this large QP problem into a series of smallest possible

QP problems. These small QP problems are solved analytically, which avoids using a time-consuming numerical QP optimization as an inner loop. SMO dramatically reduces the computational time of training SVM. Recently Scholkopf[2] proposed a variant SMO algorithm for solving the one-class support vector machine, this algorithm is an extension of the SMO algorithm proposed by Platt. In this paper, we make some improvement to SMO algorithm for OCSVM, the improvements enhance the value of SMO for OCSVM even further. Computational comparison on datasets shows that the modified version performs significantly better than the original SMO.

The paper is organized as follows. In section 2 we briefly discuss the OCSVM problem formulation. Section 3 gives a derivation of our SMO algorithm for OCSVM in detail. In section 4 we take examples to demonstrate the learning performance of the proposed method and compare against the original SMO. Finally, a conclusion and outlook can be found in section 5

2. One-Class Support Vector Machine

Given n examples, the specific optimization for estimating the hyperplane specified by the hyperplane's normal vector in the feature space ω and offset from the origin ρ is:

$$\min_{\omega\in F,\xi\in R^l,\rho\in R} \quad \frac{1}{2}\left\|\omega\right\|^2+\frac{1}{\upsilon\ell}\sum_{i=1}^{\ell}\xi_i-\rho \qquad (1)$$

$$s.t. \quad (\omega\cdot\Phi(x_i))\ge\rho-\xi_i, \quad \xi_i\ge 0 \qquad (2)$$

Where $0<\upsilon<1$ is a parameter that controls the trade-off between maximizing the distance from the origin and containing most of the data in the region created by the hyperplane and corresponds to the ratio of expected anomalies in the data set. ξ_i are slack variables that penalize the objective function but allow some of the points to be on the other wrong side of the hyperplane.

After we solve the optimization problem, the decision function for each point x is:

$$f(x)=\operatorname{sgn}((\omega\cdot\Phi(x))-\rho) \qquad (3)$$

Using multipliers $\alpha_i,\beta_i\ge 0$, we introduce a Lagrangian:

$$L(\omega,\xi,\rho,\alpha,\beta)$$
$$=\frac{1}{2}\left\|\omega\right\|^2+\frac{1}{\upsilon\ell}\sum_{i=1}^{\ell}\xi_i-\rho-\sum_{i=1}^{\ell}\alpha_i((\omega\cdot\Phi(x_i))-\rho+\xi_i)-\sum_{i=1}^{\ell}\beta_i\xi_i \qquad (4)$$

and set the derivatives with respect to the primal variables ω,ξ,ρ equal to zero, yielding:

$$\frac{dL(\omega,\xi,\rho,\alpha,\beta)}{d\omega}=\omega-\sum_{i=1}^{\ell}\alpha_i\Phi(x_i)=0$$
(5)

$$\frac{dL(\omega,\xi,\rho,\alpha,\beta)}{d\xi_i}=\frac{1}{\upsilon\ell}-\alpha_i-\beta_i=0 \qquad (6)$$

$$\frac{dL(\omega,\xi,\rho,\alpha,\beta)}{d\rho}=-1+\sum_{i=1}^{\ell}\alpha_i=0 \qquad (7)$$

From the equation(5),(6),(7), we can get:

$$\omega=\sum_{i=1}^{\ell}\alpha_i\Phi(x_i) \qquad (8)$$

$$\alpha_i=\frac{1}{\upsilon\ell}-\beta_i \qquad (9)$$

$$\sum_{i=1}^{\ell}\alpha_i=1 \qquad (10)$$

Substituting equation (8) and equation (9) into equation (4), and using kernel function method:

$$k(x,y)=(\Phi(x)\cdot\Phi(y)) \qquad (11)$$

We obtain the dual problem:

$$\frac{1}{2}\sum_{i,j=1}^{\ell}\alpha_i\alpha_j k(x_i,x_j) \qquad (12)$$

$$s.t. \quad \begin{array}{l} 0\le\alpha_i\le\dfrac{1}{\upsilon\ell} \\ \displaystyle\sum_{i=1}^{\ell}\alpha_i=1 \end{array} \qquad (13)$$

In formula (8), all patterns $\{x_{\square} : i\in[\ell],\alpha_i>0\}$ are called support vectors. Together with formula(11), the SV expansion transforms the decision function, formula (3) into a kernel expansion:

$$f(x)=\operatorname{sgn}(\sum_{i=1}^{i}\alpha_i k(x_i,x)-\rho) \qquad (14)$$

One can show that at the optimum, the two inequality constraints, equation (2), become equalities if α_i and β_i are nonzero. Therefore, we can recover ρ by exploiting that for any such α_i, the corresponding pattern x_i satisfies:

$$\rho=(\omega\cdot\Phi(x_i))=\sum_{j=1}^{\ell}\alpha_j k(x_j,x_i) \qquad (15)$$

3. A Fast SMO Algorithm for OCSVM

3.1 Solve for the Two QP Problem

SMO break a large QP problem into a series of size two QP problems. The analytical solution to the size two QP problem must be solved. The bulk of this section will be devoted to deriving this solution.

For brevity, we use k_{ij} denote $k(x_i,x_j)$. Our goal is to analytically express the minimum of (12) as a function of two parameters. Let these two parameters have indices a and b so that α_a, α_b are the two unknown parameters. We can rewrite the objective function of Formula (12) as:

$$L(\alpha_a, \alpha_b) = \frac{1}{2}\alpha_a^2 k_{aa} + \frac{1}{2}\alpha_b^2 k_{bb} + \alpha_a \alpha_b k_{ab} + \sum_{i=a,b} \alpha_i \sum_{j=1, j\neq a,b}^{\ell} \alpha_j k_{ij} + L' \quad (16)$$

Where L' is a term that is strictly constant with respect to α_a, α_b, Note that a superscript * is used below to explicitly indicate that values are computed with the old parameters values.

Due to the equality constraints of optimization problem, we let:

$$s^* = \alpha_a^* + \alpha_b^* = \alpha_a + \alpha_b$$

(17)

from(17), we get:

$$\alpha_a = s^* - \alpha_b \quad (18)$$

Substituting（18）into（16）:

$$L(\alpha_b) = \frac{1}{2}(s^* - \alpha_b)^2 k_{aa} + \frac{1}{2}\alpha_b^2 k_{bb} + (s^* - \alpha_b)\alpha_b k_{ab} + \alpha_b \sum_{j=1, j\neq a,b}^{\ell} \alpha_j k_{bj}$$

$$+(s^* - \alpha_b) \sum_{j=1, j\neq a,b}^{\ell} \alpha_j k_{aj} + L' \quad (19)$$

To solve the optimization value of equation (19), we need to compute its partial derivative with respect to α_b, i.e., $\partial L(\alpha_b)/\partial \alpha_b$:

$$\frac{\partial L(\alpha_b)}{\partial \alpha_b} = (\alpha_b - s^*)k_{aa} + \alpha_b k_{bb} + (s^* - 2\alpha_b)k_{ab} + \sum_{j=1, j\neq a,b}^{\ell} \alpha_j k_{bj} - \sum_{j=1, j\neq a,b}^{\ell} \alpha_j k_{aj} \quad (20)$$

Now, by $\frac{\partial L(\alpha_b)}{\partial \alpha_b} = 0$, we get:

$$(\alpha_b - s^*)k_{aa} + \alpha_b k_{bb} + (s^* - 2\alpha_b)k_{ab} + \sum_{j=1, j\neq a,b}^{\ell} \alpha_j k_{bj} - \sum_{j=1, j\neq a,b}^{\ell} \alpha_j k_{aj} = 0 \quad (21)$$

From equation (21), we can write an update rule for α_b:

$$\alpha_b = \eta[s^*(k_{aa} - k_{ab}) - \sum_{j=1, j\neq a,b}^{\ell} \alpha_j k_{bj} + \sum_{j=1, j\neq a,b}^{\ell} \alpha_j k_{aj}] \quad (22)$$

where $\eta = \frac{1}{k_{aa} + k_{bb} - 2k_{ab}}$.

Let:

$$\bar{f}(x) = \sum_{i=1}^{\ell} \alpha_i k(x_i, x) - \rho \quad (23)$$

$$\alpha_b = \eta[s^*(k_{aa} - k_{ab}) - \bar{f}^*(b) + \bar{f}^*(a) + (s^* - \alpha_b^*)k_{ba} + \alpha_b^* k_{bb} - (s^* - \alpha_b^*)k_{aa} - \alpha_b^* k_{ab}]$$

$$= \eta[\bar{f}^*(a) - \bar{f}^*(b) + \alpha_b^*(k_{bb} + k_{aa} - 2\alpha_b^*)]$$

$$= \alpha_b^* + \eta[\bar{f}^*(a) - \bar{f}^*(b)] \quad (24)$$

Formula (24) is an update rule for α_b in terms of its old value.

We also need to consider how the linear and boxed constraints relate to one another. In particular, we need lower and upper bounds for α_b, α_a that insure that $0 \le \alpha_b \le \frac{1}{\upsilon\ell}, 0 \le \alpha_a \le \frac{1}{\upsilon\ell}$. Using:

$$L = \max(s^* - \frac{1}{\upsilon\ell}, 0)$$
$$H = \min(\frac{1}{\upsilon\ell}, s^*) \quad (25)$$

with L and H being the lower and upper bounds, respectively, guarantees that both parameters will obey the boxed constraints.

3.2 Working Pair Selection Strategy

SMO usually uses heuristics to choose which two Lagrange multipliers to jointly optimize, so the method to choose two Lagrange multipliers for optimization at each step is crucial for SMO algorithm, In order to improve the convergence rate, we propose a new heuristic method to choose two Lagrange multipliers.

The Karush-Kuhn-Tucker conditions can be exploited to check the optimality of the current solution. Before each optimization step, the KKT conditions for α_i are evaluated to decide if an update of α_i is needed. This check takes only very few computation time if an function cache is used. The conditions for α_i are:

$$\alpha_i((\omega \cdot \Phi(x_i)) - \rho + \xi_i) = 0$$
$$\beta_i \xi_i = 0$$

At the optimal solution, the KKT conditions can be fulfilled in three different ways:

Case 1: $\alpha_i = 0 \Rightarrow \beta_i = \frac{1}{\upsilon\ell} - \alpha_i = \frac{1}{\upsilon\ell} \neq 0 \Rightarrow \xi_i = 0$
$\Rightarrow \omega \cdot \Phi(x_i)) - \rho + \xi_i > 0 \Rightarrow \omega \cdot \Phi(x_i)) - \rho > 0 \Rightarrow$
$\bar{f}(x_i) > 0$

Case2: $0 < \alpha_i < \frac{1}{\upsilon\ell} \Rightarrow \beta_i = \frac{1}{\upsilon\ell} - \alpha_i = \frac{1}{\upsilon\ell} \neq 0 \Rightarrow \xi_i = 0$
$\Rightarrow \omega \cdot \Phi(x_i)) - \rho + \xi_i = 0 \Rightarrow \omega \cdot \Phi(x_i)) - \rho = 0 \Rightarrow$
$\bar{f}(x_i) = 0$

Case3: $\alpha_i = \frac{1}{\upsilon\ell} \Rightarrow \beta_i = \frac{1}{\upsilon\ell} - \alpha_i = 0 \Rightarrow \xi_i \neq 0$
$\Rightarrow \omega \cdot \Phi(x_i)) - \rho + \xi_i = 0 \Rightarrow \omega \cdot \Phi(x_i)) - \rho < 0 \Rightarrow a$
$\bar{f}(x_i) < 0$

From the above mentioned, an optimal α_i has to obey one of the following three conditions (KKT condition):

$\alpha_i = 0 \quad \wedge \quad \bar{f}(x_i) > 0$

$0 < \alpha_i < \frac{1}{\upsilon\ell} \quad \wedge \quad \bar{f}(x_i) = 0$

$\alpha_i = \frac{1}{\upsilon\ell} \quad \wedge \quad \bar{f}(x_i) < 0$

For the dual optimal problem(12), when we only consider its equality constraint condition, we may get the following Langrage function:

$$\bar{L} = \frac{1}{2}\sum_{i,j=1}^{\ell}\alpha_i\alpha_j k(x_i, x_j) + \lambda(1 - \sum_{i=1}^{\ell}\alpha_i) \tag{26}$$

According to [7], Lagrange multiplier λ equals to the parameter ρ in formula (23), namely:

$\lambda = \rho$

So we can get :

$$\bar{L} = \frac{1}{2}\sum_{i,j=1}^{\ell}\alpha_i\alpha_j k(x_i, x_j) + \rho(1 - \sum_{i=1}^{\ell}\alpha_i) \tag{27}$$

For (27), we solve objective function's gradient for α_b:

$$\frac{\partial\bar{L}}{\partial\alpha_b} = \sum_{i=1}^{\ell}\alpha_i k_{ib} - \rho = \bar{f}(x_b) \tag{28}$$

For an objective function, the bigger its gradient's absolute value for a variable, the bigger its variation when the optimal variable change. So according to this point we choose the first optimal variable. For the other optimal variables choose, just like Scholkopf's method, to make the variable's variation the biggest. According to above statement, the strategy to working pair selection is as following: firstly select the first variable α_b which makes $Max(|\bar{f}(x_b)|)$ holding, then choose the second variable α_a which makes $Max(|\bar{f}(x_b) - \bar{f}(x_a)|)$ holding. The concrete choosing steps are as following: There are two separate choice heuristics: one for $\bar{\alpha}_b$ and one for $\bar{\alpha}_a$. The outer loop of the algorithm first iterates over the entire set of training examples deciding whether an example violates the KKT condition and makes $Max(|\bar{f}(x_b)|)$ holding, if it does, then that example responding variable is chosen for optimization immediately, if it does not, stop program and output results; a second variable is found by maximizing the step that can be taken during joint optimization, namely makes $Max(|\bar{f}(x_b) - \bar{f}(x_a)|)$ holding, then the two multipliers are jointly optimized.

3.3 The Complete Algorithm

From the above mentioned, the whole SMO algorithm is as follow:

1. initialize $\alpha_a^{old}, \alpha_b^{old}$
2. choose two parameters, α_a, α_b by 3.2.
3. compute α_b^{new} by Formula (24)
4. If $\alpha_b^{new} > H$, set $\alpha_b^{new} = H$, if $\alpha_b^{new} < L$, set $\alpha_b^{new} = L$;
5. $\alpha_a^{new} = \alpha_a^{old} + \alpha_b^{old} - \alpha_b^{new}$;
6. compute ρ by Formula (15)
7. if satisfy the KKT stopping criterion then go to 8 else go to 2
8. output α , stop program.

4. Simulation and Comparisons

In this section we will take concrete examples to illustrate the proposed method. We take Gaussian function as kernel, namely $k(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2})$, and give out SVM's design parameter as : $\sigma = 1.5$ $\upsilon = 0.4$ .We apply the SMO algorithm to artificial data. At the same time we compare the performance of the proposed SMO algorithm and the original SMO algorithm proposed by Scholkopf. The simulating figures and table are as follows:

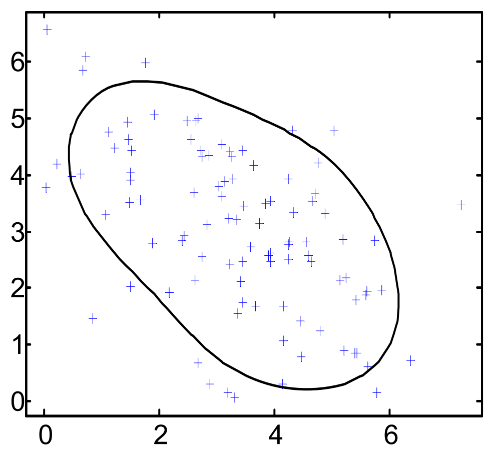

Figure 2. The simulation result of 100 samples

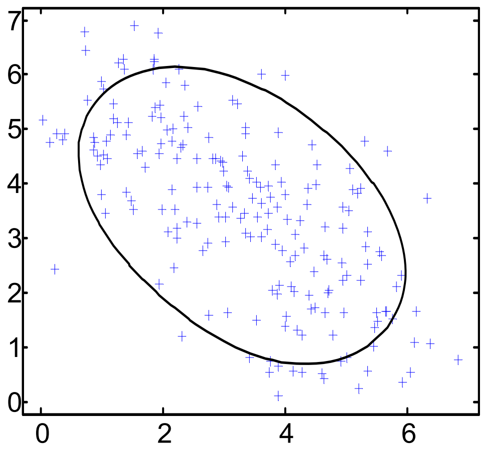

Figure 3. The simulation result of 100 samples

Table 1. The running time of original SMO and improved SMO

Sample number		100	200	500	800	1000	2000	3000
Improved SMO	Running time	0.297	0.643	1.417	5.254	9.648	57.325	216.852
Original SMO	Running time	0.75	1.265	3.704	11.907	23.641	147.859	582.813

Both trained SVM's training and testing error are almost the same, but their running time are very different, The proposed algorithm is several times faster than the original SMO.

5. Conclusion and Outlook

In this paper, we proposed an improved SMO algorithm to train one-class support vector machine, in which a new working set selection strategy was adopted to improve algorithm's converging rate. The experimental results presented show that our proposed learning techniques are capable of achieving performance results similar to the original SMO algorithm, while improving the training time. Future work will concentrate on designing incremental SMO algorithm, and use it to solve the learning problem of sequential data.

References

[1] Vapnik,V.N., "The nature of statistical learning theory", Springer, New York, 1995

[2] Scholkopf, B., Platt, J.C., Taylor, J.S., "Estimating the support of a high dimensional distribution Neural Computation", 2001, pp1443–1471

[3] Cristianini, N., Shawe-Taylor, J., "An Introduction to Support Vector Machines", Cambridge University Press, Cambridge, UK, 2000

[4] Stolfo, S., Wang, K., "One class training for masquerade detection", 3rd IEEE Conference Data Mining Workshop on Data Mining for Computer Security, Florida, 2003

[5] Yunqiang, C.H., Xiang, Z.H., and Thomas, S., "One-class svm for learning in image retrieval", Proceedings of IEEE International Conference on Image Processing, Thessaloniki, Greece, 2001

[6] Platt, J.C., "Fast training of support vector machines using sequential minimal optimization", In: Scholkopf, B., Burges, C., Smola, A.: Advances in Kernel Methods: Support vector Machines, MIT Press. Cambridge, MA, 1998

[7] Martin,M., "On-line Support Vector Machines for Function Approximation", Technical Report LSI-02-11-R, Software Department, Universitat Politecnica de Catalunya, 2002

IEEE COMPUTER SOCIETY