2005 International Symposium on Nonlinear
Theory and its Applications (NOLTA2005)
Bruges, Belgium, October 18-21, 2005

An Efficient Method for Searching Optimal Kernel Parameter of Support Vector Machines

Keisuke Arima† and Norikazu Takahashi†

†Department of Computer Science and Communication Engineering, Kyushu University
6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581 Japan
Email: arima@kairo.csce.kyushu-u.ac.jp, norikazu@csce.kyushu-u.ac.jp

Abstract—Generalization capability of support vector machines depends heavily on the kernel function and its parameters. In this paper, we focus our attention on the Gaussian kernel and propose an efficient method for finding the kernel parameter which minimizes the number of support vectors. Since the generalization error estimated by the leave-one-out procedure is upper-bounded by the ratio of the number of support vectors to the number of training samples, the proposed method will be useful for finding support vector machines with higher generalization capability.

1. Introduction

Support vector machines (SVMs) [1, 2] have recently attracted great attention in the fields of pattern recognition, machine learning, neural networks, signal processing, and so on. An SVM maps input patterns into the feature space with a kernel function and then classifies them into two classes by a hyperplane in the feature space. Since generalization capability of an SVM depends heavily on the kernel function, it is important to find the optimal kernel function efficiently. So far, various techniques for finding the kernel parameter for which smaller generalization error is achieved have been proposed [3]. Many of them is based on leave-one-out (LOO) estimate of the generalization error because LOO error has good properties. However, LOO estimate is time-consuming.

In this paper, we focus our attention on the Gaussian kernel, the most widely used kernel for SVMs, and consider the problem of finding the kernel parameter which minimizes the number of support vectors (SVs). Since LOO estimate of the generalization error is upper-bounded by the ratio of the number of SVs to the number of training samples [3], this problem is important for increasing the generalization capability of SVMs. In fact, as shown in Fig. 1, the kernel parameter minimizing the number of SVs is very close to that minimizing the generalization error. In addition, since the number of SVs represents the simplicity of the decision function of SVMs, this problem is also important for speeding up the classification time for test patterns.

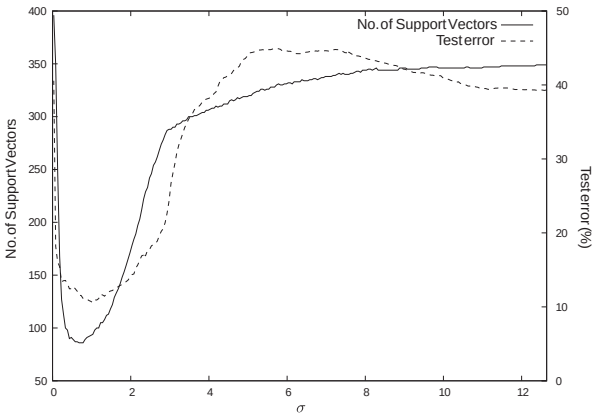

Figure 1: Dependence of the generalization error and the number of SVs on the kernel parameter.

2. Problem Formulation

Suppose that we are given a set of l training samples $\{(\boldsymbol{x}_i, d_i)\}_{i=1}^{l}$ where $\boldsymbol{x}_i \in \mathbb{R}^n$ is the i-th input pattern and $d_i \in \{1, -1\}$ represents the class to which $\boldsymbol{x}_i$ belongs. The learning of an SVM with the kernel function $K(\cdot, \cdot)$ leads to the following quadratic programming (QP) problem [1].

Problem 1 *Find* $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_l]^T$ *which minimizes the objective function*

$$W(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} q_{ij}\alpha_i\alpha_j - \sum_{i=1}^{l}\alpha_i$$

under the constraints

$$\sum_{i=1}^{l} d_i\alpha_i = 0 \tag{1}$$

$$0 \le \alpha_i \le C, \quad i = 1, 2, \ldots, l \tag{2}$$

where $q_{ij} = d_i d_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ *and* C *is a user-specified positive constant.*

If the kernel function $K(\cdot, \cdot)$ satisfies the Mercer's condition, the matrix $\boldsymbol{Q} = [q_{ij}]$ becomes positive semi-definite and therefore Problem 1 has no local minimum [1]. Let

$\boldsymbol{\alpha}^* = [\alpha_1^*, \alpha_2^*, \ldots, \alpha_l^*]^T$ be an optimal solution of Problem 1 and i_0 be any i such that $0 < \alpha_i^* < C$. Then the decision function of the SVM can be expressed as

$$f(\boldsymbol{x}) = \mathrm{sgn}\left(\sum_{i=1}^{l} \alpha_i^* d_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b\right) \tag{3}$$

where b is given by

$$b = d_{i_0} - \sum_{i=1}^{l} \alpha_i^* d_i K(\boldsymbol{x}_i, \boldsymbol{x}_{i_0}) \,. \tag{4}$$

Since only a small number of α_i^*'s take nonzero values in general, the decision function (3) is composed of a small number of terms. The input pattern $\boldsymbol{x}_i$ corresponding to a nonzero α_i^* is called a support vector.

In this paper, we restrict ourselves to the Gaussian kernel defined by

$$K(\boldsymbol{x}, \boldsymbol{x}') = K_\sigma^{\mathrm{G}}(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{||\boldsymbol{x} - \boldsymbol{x}'||^2}{2\sigma^2}\right)$$

where σ is a positive number determined by users. Also, we assume hereafter that all $\boldsymbol{x}_i$'s are distinct. Then $\boldsymbol{Q} = [q_{ij}]$ becomes positive definite for any $\sigma > 0$ [2] which implies that Problem 1 has a unique optimal solution.

The problem considered in this paper is as follows:

Problem 2 *Find σ which minimizes the objective function*

$$V(\sigma) = |\{i \,|\, \alpha_i^* > 0\}|$$

where $\boldsymbol{\alpha}^* = [\alpha_1^*, \alpha_2^*, \ldots, \alpha_l^*]^T$ is the optimal solution of Problem 1 with $K(\cdot, \cdot) = K_\sigma^{\mathrm{G}}(\cdot, \cdot)$.

Problem 2 is known as a bilevel optimization problem such that the first level problem is to minimize $V(\sigma)$ and the second level is Problem 1. It is known that bilevel optimization problems are difficult to solve even if both the first and second level problems are linear. In our case, the first level problem is a discrete optimization and the second level is a QP problem. Therefore, Problem 2 is thought to be extremely hard to solve.

In the following, in order to make clear the dependence of the matrix $\boldsymbol{Q}$ and the optimal solution $\boldsymbol{\alpha}^*$ of Problem 1 on the parameter σ, we will use the notations $\boldsymbol{Q}(\sigma) = [q_{ij}(\sigma)]$ and $\boldsymbol{\alpha}^*(\sigma) = [\alpha_1^*(\sigma), \alpha_2^*(\sigma), \ldots, \alpha_l^*(\sigma)]^T$.

3. Preliminaries

In this section, we will present some basic results which will be useful for the proposed method.

3.1. Optimality Condition

Since $\boldsymbol{Q}$ is positive definite, an $\boldsymbol{\alpha} \in S$, where S denotes the feasible region of Problem 1, is the optimal solution if and only if the Karush-Kuhn-Tucker (KKT) conditions are satisfied. Moreover, the KKT conditions can be rewritten in a compact form as follows [4]:

$$\min_{i \in I_{\mathrm{up}}(\boldsymbol{\alpha})} F_i(\boldsymbol{\alpha}) \geq \max_{i \in I_{\mathrm{low}}(\boldsymbol{\alpha})} F_i(\boldsymbol{\alpha}) \tag{5}$$

where

$$\begin{aligned} F_i(\boldsymbol{\alpha}) &= d_i\left(\sum_{j=1}^{l} q_{ij}(\sigma)\alpha_j - 1\right), \\ I_{\mathrm{up}}(\boldsymbol{\alpha}) &= \{i \,|\, 0 < \alpha_i < C\} \cup \{i \,|\, \alpha_i = 0, d_i = 1\} \\ &\quad \cup \{i \,|\, \alpha_i = C, d_i = -1\}, \\ I_{\mathrm{low}}(\boldsymbol{\alpha}) &= \{i \,|\, 0 < \alpha_i < C\} \cup \{i \,|\, \alpha_i = 0, d_i = -1\} \\ &\quad \cup \{i \,|\, \alpha_i = C, d_i = 1\}. \end{aligned}$$

In a practical situation, the optimality condition (5) is often relaxed as

$$\min_{i \in I_{\mathrm{up}}(\boldsymbol{\alpha})} F_i(\boldsymbol{\alpha}) \geq \max_{i \in I_{\mathrm{low}}(\boldsymbol{\alpha})} F_i(\boldsymbol{\alpha}) - \tau \tag{6}$$

where τ is a positive constant [4,5]. A pair of indices (i, j) such that

$$i \in I_{\mathrm{up}}(\boldsymbol{\alpha}),\ j \in I_{\mathrm{low}}(\boldsymbol{\alpha}),\ F_i(\boldsymbol{\alpha}) \leq F_j(\boldsymbol{\alpha}) - \tau$$

is called a τ-violating pair at $\boldsymbol{\alpha}$. It is obvious from these definitions that $\boldsymbol{\alpha} \in S$ satisfies the optimality condition (6) if and only if there is no τ-violating pair at $\boldsymbol{\alpha}$.

3.2. Continuity of the Optimal Solution

Since the second level problem of Problem 2 is parameterized with σ, it is important to check whether the solution $\boldsymbol{\alpha}^*(\sigma)$ is continuous with respect to σ or not. This question is positively solved by the following theorem.

Theorem 1 *The optimal solution $\boldsymbol{\alpha}^*(\sigma)$ of Problem 1 is continuous with respect to σ.*

Proof: Let σ_0 be any positive number. Let $\epsilon = ||\boldsymbol{Q}(\sigma) - \boldsymbol{Q}(\sigma_0)||$. Then it follows from Theorem 2.1 of [6] that

$$||\boldsymbol{\alpha}^*(\sigma) - \boldsymbol{\alpha}^*(\sigma_0)|| \leq \frac{\epsilon}{\lambda - \epsilon} ||\boldsymbol{\alpha}^*(\sigma_0)|| \tag{7}$$

holds if $\epsilon < \lambda$ where λ is the smallest eigenvalue of $\boldsymbol{Q}(\sigma_0)$. Since the matrix $\boldsymbol{Q}(\sigma)$ is apparently continuous at $\sigma = \sigma_0$, ϵ goes to zero as σ approaches σ_0. Hence the right-hand side of (7) converges to 0 as σ approaches σ_0. This implies that $\boldsymbol{\alpha}^*(\sigma)$ is continuous at σ_0. □

Theorem 1 says that if σ is sufficiently close to σ_0 then $\boldsymbol{\alpha}^*(\sigma)$ exists in the neighborhood of $\boldsymbol{\alpha}^*(\sigma_0)$. This property plays an important role in the proposed method.

4. Proposed Method

4.1. How to Update σ

The curve of $V(\sigma)$ often has one deep valley as shown in Fig.1. However, since $V(\sigma)$ takes discrete values, it is

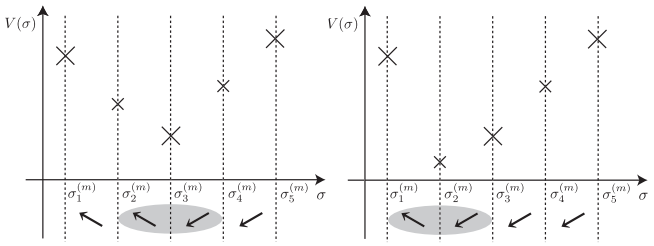

Figure 2: How to update the interval.

impossible to define the derivative of $V(\sigma)$ with respect to σ. In fact, there are a number of local peaks and valleys as one can see in Fig.1. Therefore traditional line search algorithms are not applicable to this problem. So we will employ the following strategy: First, we set σ to a sufficiently large value σ_0 and then moves to the left with a fixed step size h as $\sigma_k = \sigma_0 - kh$ $(k = 0, 1, 2, \ldots)$ until

$$V(\sigma_k) > V(\sigma_{k-1}) \quad \text{and} \quad V(\sigma_{k-1}) < V(\sigma_{k-2})$$

are satisfied. It is apparent that the minimum point necessarily belongs to the interval $I_1 = [\sigma_k, \sigma_{k-2}]$ of length $2h$ if there is only one valley. Next we set $\sigma_1^{(1)} = \sigma_k$, $\sigma_2^{(1)} = (\sigma_k + \sigma_{k-1})/2$, $\sigma_3^{(1)} = \sigma_{k-1}$, $\sigma_4^{(1)} = (\sigma_{k-1} + \sigma_{k-2})/2$ and $\sigma_5^{(1)} = \sigma_{k-2}$, and then find $n \in \{1, 2, 3\}$ such that

$$V(\sigma_n^{(1)}) > V(\sigma_{n+1}^{(1)}) \quad \text{and} \quad V(\sigma_{n+1}^{(1)}) < V(\sigma_{n+2}^{(1)})$$

are satisfied. Once again, the minimum point belongs to the interval $I_2 = [\sigma_n^{(1)}, \sigma_{n+2}^{(1)}]$ of length h if there is only one valley (see Fig.2). By repeating this process, we can obtain the sequence of intervals $I_1, I_2, \ldots$ such that the length of I_m is given by $h(1/2)^{m-1}$ and the minimum point must belong to all of them. The process is stopped when the interval becomes sufficiently short.

This strategy is formally expressed as follows:

Algorithm 1 Given the training samples $\{(\boldsymbol{x}_i, d_i)\}_{i=1}^l$, the parameter C, the initial value σ_0 of σ, the initial step size h and the minimum step size $h_{\min}$, find σ^* by executing the following procedures.

1) Set $k = 0$, $\sigma_1 = \sigma_0 - h$ and $\sigma_2 = \sigma_0 - 2h$.

2) If $V(\sigma_{k+2}) > V(\sigma_{k+1})$ and $V(\sigma_{k+1}) < V(\sigma_k)$ hold, then go to Step 4). Otherwise go to Step 3).

3) Add 1 to k, set $\sigma_{k+2} = \sigma_k - 2h$, and go to Step 2).

4) Set $m = 1$ and

$$\begin{array}{ll} \sigma_1^{(m)} = \sigma_{k+2}, & \sigma_2^{(m)} = (\sigma_{k+2} + \sigma_{k+1})/2, \\ \sigma_3^{(m)} = \sigma_{k+1}, & \sigma_4^{(m)} = (\sigma_{k+1} + \sigma_k)/2, \\ \sigma_5^{(m)} = \sigma_k. & \end{array}$$

5) If $\sigma_5^{(m)} - \sigma_1^{(m)} < h_{\min}$ then set $\sigma^* = \sigma_3^{(m)}$ and stop. Otherwise go to Step 6).

6) Find $n \in \{1, 2, 3\}$ such that $V(\sigma_n^{(m)}) > V(\sigma_{n+1}^{(m)})$ and $V(\sigma_{n+1}^{(m)}) < V(\sigma_{n+2}^{(m)})$ are satisfied. Add 1 to m, set

$$\begin{array}{ll} \sigma_1^{(m)} = \sigma_n^{(m-1)}, & \sigma_2^{(m)} = (\sigma_n^{(m-1)} + \sigma_{n+1}^{(m-1)})/2, \\ \sigma_3^{(m)} = \sigma_{n+1}^{(m-1)}, & \sigma_4^{(m)} = (\sigma_{n+1}^{(m-1)} + \sigma_{n+2}^{(m-1)})/2, \\ \sigma_5^{(m)} = \sigma_{n+2}^{(m-1)}, & \end{array}$$

and go to Step 5).

4.2. How to Find $\boldsymbol{\alpha}^*(\sigma)$

The most time-consuming task in Algorithm 1 is to find the optimal solution $\boldsymbol{\alpha}^*(\sigma)$ for various values of σ. So it is important to reduce the computation time required for this part. Here we will propose a method of computing $\boldsymbol{\alpha}^*(\sigma)$ with high efficiency.

The key idea is to make use of the previously obtained values of $\boldsymbol{\alpha}^*(\sigma)$. Let us explain this by considering $\boldsymbol{\alpha}^*(\sigma_2^{(m)})$ in Step 6). Since $\sigma_2^{(m)}$ is close to $\sigma_1^{(m)}$ and $\sigma_3^{(m)}$ for large m, it is expected from Theorem 1 that $\boldsymbol{\alpha}^*(\sigma_2^{(m)})$ exists in the neighborhood of $\boldsymbol{\alpha}^*(\sigma_1^{(m)})$ or $\boldsymbol{\alpha}^*(\sigma_3^{(m)})$, where both $\boldsymbol{\alpha}^*(\sigma_1^{(m)})$ and $\boldsymbol{\alpha}^*(\sigma_3^{(m)})$ have already been obtained in the process of determining n. Therefore, when we try to find $\boldsymbol{\alpha}^*(\sigma_2^{(m)})$ by solving Problem 1, the computation time will be reduced if we set the initial feasible solution to $(\boldsymbol{\alpha}^*(\sigma_1^{(m)}) + \boldsymbol{\alpha}^*(\sigma_3^{(m)}))/2$ instead of the zero vector[1]. Moreover, it is also expected that there exist only a small number of τ-violating pairs at the initial feasible solution $(\boldsymbol{\alpha}^*(\sigma_1^{(m)}) + \boldsymbol{\alpha}^*(\sigma_3^{(m)}))/2$. The computation time will be further reduced by using the decomposition method [5] which is a well-known iterative technique to solve large QP problems.

The proposed method is formally stated as follows:

Algorithm 2 Given σ, find $\boldsymbol{\alpha}^*(\sigma)$ by executing the following procedures.

1) Set $u = 0$ and determine the initial feasible solution $\boldsymbol{\alpha}_0$. When $\boldsymbol{\alpha}^*(\sigma_k)$ has to be found in Step 2) of Algorithm 1, $\boldsymbol{\alpha}_0$ is set to $\boldsymbol{\alpha}^*(\sigma_{k-1})$ if $k \neq 0$ and the zero vector if $k = 0$. When $\boldsymbol{\alpha}^*(\sigma_k^{(m)})$ $(k = 2$ or $4)$ has to be found in Step 6) of Algorithm 1, $\boldsymbol{\alpha}_0$ is set to $(\boldsymbol{\alpha}^*(\sigma_{k-1}^{(m)}) + \boldsymbol{\alpha}^*(\sigma_{k+1}^{(m)}))/2$.

2) If (6) is satisfied with $\boldsymbol{\alpha} = \boldsymbol{\alpha}_u$, then set $\boldsymbol{\alpha}^*(\sigma) = \boldsymbol{\alpha}_u$ and stop. Otherwise go to Step 3).

3) Find all τ-violating pairs at $\boldsymbol{\alpha}_u$. Let $B \subseteq \{1, 2, \ldots, l\}$ be the set of all i such that (i, j) or (j, i) is a τ-violating pair at $\boldsymbol{\alpha}_u$ for some j.

4) Solve Problem 1 under the additional constraints $\alpha_i = \alpha_{u,i}$ for all $i \notin B$. Set $\boldsymbol{\alpha}_{u+1}$ to the optimal solution, add 1 to k, and go to Step 2).

[1] In QP problem solvers such as `quadprog` of Matlab and `quapro` of Scilab, users can optionally set the initial feasible solution.

We call the optimization problem in Step 4) the subproblem. Note that the subproblem is a QP problem with the variables α_i, $i \in B$. If $\boldsymbol{\alpha}_u$ is close to the optimal solution, the number of τ-violating pairs at $\boldsymbol{\alpha}_u$ is small and therefore the size of the subproblem is also small. Hence each subproblem can be solved much faster than the original QP problem having l variables. This is the main advantage of the decomposition method.

5. Experimental Results

In order to verify the efficiency of the proposed method, we have carried out some experiments with the benchmark data[2]. In the experiments, the parameter C was set to the values specified at the web site of the benchmark data except for "diabetis"[3]. The initial value of σ was set to

$$\sigma_0 = \max_{i,j} ||\boldsymbol{x}_i - \boldsymbol{x}_j|| / \sqrt{-2 \log 0.9}$$

so that $\min_{ij} |q_{ij}(\sigma_0)| = 0.9$ holds. The initial and minimum step sizes for σ are set to $h = \sigma_0/20$ and $h_{\min} = \sigma_0/2^8$, respectively.

For the purpose of comparison, we have used the following three methods in addition to the proposed method. All of these methods are implemented in Scilab.

1) Method 1: The parameter σ is updated with the fixed step size $h_{\min}$ as $\sigma_k = \sigma_0 - k h_{\min}$ $(k = 0, 1, 2, \ldots)$. For each k, $\boldsymbol{\alpha}^*(\sigma_k)$ is found by executing `quapro` once without specifying the initial feasible solution. The value of σ_k which minimizes $V(\sigma_k)$ is set to σ^*.

2) Method 2: The parameter σ is updated as in Method 1. For each k, $\boldsymbol{\alpha}^*(\sigma_k)$ is found by executing `quapro` once with the initial feasible solution $\boldsymbol{\alpha}^*(\sigma_{k-1})$. The value of σ_k which minimizes $V(\sigma_k)$ is set to σ^*.

3) Method 3: The parameter σ is updated as in Method 1. For each k, $\boldsymbol{\alpha}^*(\sigma_k)$ is found by Algorithm 2. The value of σ_k which minimizes $V(\sigma_k)$ is set to σ^*.

Experimental results are shown in Tables 1 and 2. Table 1 shows the CPU time spent by each method. By comparing the results of Methods 1 to 3, we see that the specification of the initial feasible solution and the application of the decomposition method effectively reduce the computation time. Also, by comparing the results of Method 3 and the proposed method, we can say that the strategy for updating σ in Algorithm 1 is very effective. Table 2 shows comparison results between Method 3 and the proposed method in terms of the number of SVs and test error. As shown in these results, the number of SVs and the test error for the proposed method are at the same level as Method 3, while the computation time is considerably reduced. From these observations, we can conclude that the proposed method is very useful for finding the optimal kernel parameter.

[2] http://users.rsise.anu.edu.au/~raetsch/data/

[3] For "diabetis" C was set to 30.

Table 1: Comparison of CPU time

Data Set	l	Method 1	Method 2	Method 3	Proposed
diabetis	468	1084.50	391.73	119.52	18.50
banana	400	766.54	195.34	39.31	17.17
breast-cancer	200	89.87	61.51	34.23	2.15
heart	170	32.71	21.28	17.37	1.70
thyroid	140	12.33	7.60	5.65	0.79

Table 2: Comparison of test error and the number of support vectors for "diabetis".

Set No.	Method 3			Proposed		
	σ^*	Error	SVs	σ^*	Error	SVs
1	4.204	24.33	233	5.629	25.33	238
2	4.204	23.00	238	3.179	24.00	239
3	3.947	24.33	233	3.635	24.67	235
4	3.589	26.33	223	5.230	25.33	224
5	3.635	24.67	231	4.653	24.67	238

6. Conclusion

We have proposed a method of finding the optimal kernel parameter of SVMs with the Gaussian kernel. We have shown via experiments with the benchmark data that the computation time can in fact be greatly reduced by using the proposed method.

Acknowledgments

This research was partially supported by the 21st Century COE Program 'Reconstruction of Social Infrastructure Related to Information Science and Electrical Engineering'.

References

[1] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[2] B. Schölkopf and A. J. Smola, *Learning with Kernels*. Cambridge: MIT Press, 2002.

[3] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, pp. 131–159, 2002.

[4] S. S. Keerthi and E. G. Gilbert, "Convergence of a generalized SMO algorithm for SVM classifier design," *Machine Learning*, vol. 46, pp. 351–360, 2002.

[5] T. Joachims, "Making large-scale support vector machine learning practical," in *Advances in Kernel Methods: Support Vector Machines*, B. Schölkopf, C. Burges, and A. Smola, Eds. Cambridge, MA: MIT Press, 1998.

[6] J. W. Daniel, "Stability of the solution of definite quadratic programs," *Mathematical Programming*, vol. 5, pp. 41–53, 1973.