Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Novel linear search for support vector machine parameter selection*

Hong-xia PANG†, Wen-de DONG, Zhi-hai XU, Hua-jun FENG†‡, Qi LI, Yue-ting CHEN

(*State Key Laboratory of Optical Instrumentation, Zhejiang University, Hangzhou 310027, China*)

†E-mail: malinda2568@sina.com; fenghj@zju.edu.cn

Received Jan. 5, 2011; Revision accepted Aug. 16, 2011; Crosschecked Sept. 28, 2011

Abstract: Selecting the optimal parameters for support vector machine (SVM) has long been a hot research topic. Aiming for support vector classification/regression (SVC/SVR) with the radial basis function (RBF) kernel, we summarize the rough line rule of the penalty parameter and kernel width, and propose a novel linear search method to obtain these two optimal parameters. We use a direct-setting method with thresholds to set the epsilon parameter of SVR. The proposed method directly locates the right search field, which greatly saves computing time and achieves a stable, high accuracy. The method is more competitive for both SVC and SVR. It is easy to use and feasible for a new data set without any adjustments, since it requires no parameters to set.

Key words: Support vector machine (SVM), Rough line rule, Parameter selection, Linear search, Motion prediction
doi:10.1631/jzus.C1100006 **Document code:** A **CLC number:** TP181

1 Introduction

Support vector machine (SVM) is a popular machine learning method introduced by Vapnik in 1970s, based on statistical learning and structural risk minimization. Some machine learning methods, such as neural network and the genetic algorithm, have problems of dimensionality curse, excessive learning, and local optimization. SVM effectively overcomes these difficulties (Vladimir and Vapnik, 2000; Deng and Tian, 2004) and is widely used in various fields, for example, pattern recognition (text classification (Wang and Chiang, 2007), face detection (Roohi *et al*., 2007), etc.), image classification and retrieval (Kim *et al*., 2007), system control (Suykens *et al*., 2001), function approximation (Vapnik *et al*., 1997), and time series prediction (Lau and Wu, 2008; Sapankevych and Sankar, 2009).

Due to SVM's high dependency on a proper setting of its parameters, the main issue is how to effectively find the optimal parameters. Aiming for the radial basis function (RBF) kernel, in support vector classification (SVC) we need to set penalty parameter C and kernel width γ, and in support vector regression (SVR) there is one more parameter, epsilon parameter ε, to set. The various methods to set SVM parameters can be grouped into four types: (1) grid search, such as four-step search (Po and Ma, 2002) and diamond search (Zhu and Ma, 1997); (2) direct setting of parameters using empirical formulas (Cherkassky and Ma, 2004; Cristianini *et al*., 2006); (3) metaheuristic algorithms, such as particle swarm optimization (Huang and Dun, 2008) and the genetic algorithm (Huang and Wang, 2006); and (4) others, such as linear search (Keerthi and Lin, 2003), which is based on asymptotic behavior rules of parameters, and the gradient descent method (Bengio, 2000; Chapelle *et al*., 2002). The prime grid search method, which searches all the points on the grid, achieves a high accuracy, but costs too much computing time. The direct-setting method performs rapidly but hardly guarantees accuracy. Metaheuristic algorithms do offer good results and are of good generality, but it is

‡ Corresponding author

* Project supported by the National Basic Research Program (973) of China (No. 2009CB724006) and the National Natural Science Foundation of China (No. 60977010)

still necessary to choose their own proper parameters (e.g., crossover rate, mutation rate, and initial population) for the algorithms when they are used to optimize SVM parameters.

In this paper, a novel linear search method is proposed to obtain the optimal C and γ effectively. This method requires no extra parameters to set and guarantees high accuracy and stability with less computing time. As for ε, we use an existing formula with thresholds to set it directly.

2 Brief principle of SVM

2.1 Principle of SVR

SVM is a powerful tool for solving classification and regression problems. It is used basically for binary classification. Regression problems, however, can be transformed equivalently to classification problems. The key idea of binary SVC is to map the original data from the input space to a high dimensional space using some mapping function $\psi(\boldsymbol{X})$, so that the nonlinear problem in a low dimensional space can be transformed to a linear quadratic programming problem in a high dimensional space (Vladimir and Vapnik, 2000; Deng and Tian, 2004). As shown in Fig. 1, '○' and '−' represent two types of data. The original nonlinear problem is to search for the optimal elliptic curve to separate these two types of data in the 2D space. By using $\psi(\boldsymbol{X})$ the original nonlinear problem can be transformed to a linear problem of searching for the right hyper-plane to separate the points in the 3D space. Obviously, it is easier to search for a plane than to search for a curve.

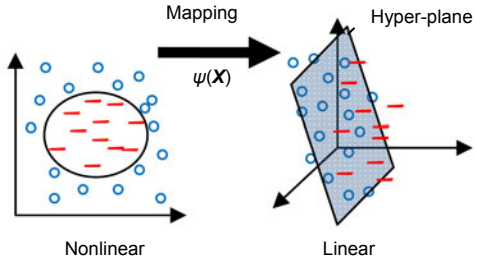

Fig. 1 Space mapping using a mapping function
'○' and '−' represent two types of data

Given a set of data $\{(\boldsymbol{X}_1, y_1), (\boldsymbol{X}_2, y_2), \ldots, (\boldsymbol{X}_n, y_n)\}$, where the vector $\boldsymbol{X}_i=(x_{i1}, x_{i2}, \ldots, x_{ik})\in\mathbb{R}^k$ (i=1, 2, …, n) is the input vector denoting k attributes of samples and has its corresponding value y_i, the problem is to obtain the objective function $f(\boldsymbol{X})$ to fit to points $(\boldsymbol{X}_i, y_i)$. Mapping function $\psi(\boldsymbol{X})$ is to transform $f(\boldsymbol{X})$ to a linear function in the feature space: $f(\boldsymbol{X})=\boldsymbol{W}\cdot\psi(\boldsymbol{X})+b$. We can directly set up the primal optimization problem of SVR:

$$\begin{aligned} &\min\ 0.5\|\boldsymbol{W}\|^2+C\sum_{i=1}^{n}\xi_i,\ C\geq 0,\\ &\text{s.t.}\ \ \boldsymbol{W}\cdot\psi(\boldsymbol{X})+b\geq y_i-\varepsilon-\xi_i,\\ &\qquad \boldsymbol{W}\cdot\psi(\boldsymbol{X})+b\leq y_i+\varepsilon+\xi_i,\\ &\qquad \xi_i\geq 0,\ i=1, 2, \ldots, n, \end{aligned} \tag{1}$$

where ξ_i is a slack variable depicting the estimation error for each point, and C is a penalty parameter punishing the total error of the training set. ε is introduced from the ε-insensitive loss function (Vladimir and Vapnik, 2000) to measure estimation quality. If the point lies inside the ε-tube (i.e., $\xi_i\leq\varepsilon$), the error will be ignored. Otherwise, it will be penalized (Fig. 2).

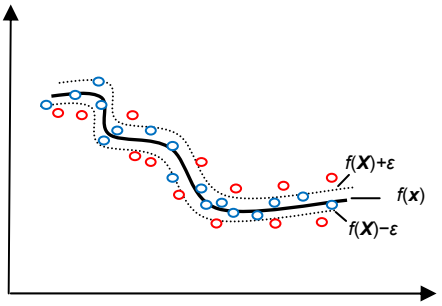

Fig. 2 ε-tube for support vector regression (SVR)
$f(\boldsymbol{X})$ is the objective function that we need to find to fit to points. The ε-tube is the area between the two dotted lines which are shifted up/down by ε from $f(\boldsymbol{X})$. Errors of points inside the ε-tube are ignored when the ε-insensitive loss function is used

The optimization Eq. (1) is usually transformed to its dual problem by introducing the Lagrange multipliers $\boldsymbol{\alpha}=(\alpha_1, \alpha_1', \alpha_2, \alpha_2', \ldots, \alpha_n, \alpha_n')$ ($0\geq\alpha_i\geq C$, $0\geq\alpha_i'\geq C$). The vector $\boldsymbol{X}_i$ corresponding to α_i ($0\geq\alpha_i\geq C$, $0\geq\alpha_i'\geq C$) are support vectors. A final decision function can be obtained:

$$f(\boldsymbol{X})=\sum_{i=1}^{n}(\alpha_i'^{*}-\alpha_i^{*})K(\boldsymbol{X}_i, \boldsymbol{X})+b. \tag{2}$$

Here the optimal Lagrange multiplier $\boldsymbol{\alpha}^{*}=(\alpha_1^{*}, \alpha_2^{*}, \ldots, \alpha_n^{*})$ is the optimization solution, and $K(\boldsymbol{X}_i, \boldsymbol{X}_j)$ is a

kernel function defined as

$$K(\boldsymbol{X}_i, \boldsymbol{X}_j) = \psi(\boldsymbol{X}_i) \cdot \psi(\boldsymbol{X}_j). \tag{3}$$

A function satisfying Mercer's condition can be accepted as a kernel function, which means its corresponding mapping function exists. Some common kernels are listed as follows (Scholkopf and Smola, 2002):

Linear kernel: $K(\boldsymbol{X}_i, \boldsymbol{X}_j)=\boldsymbol{X}_i \cdot \boldsymbol{X}_j$;
Polynomial kernel: $K(\boldsymbol{X}_i, \boldsymbol{X}_j)=(\boldsymbol{X}_i \cdot \boldsymbol{X}_j+1)^d$;
Radial basis function kernel:
$K(\boldsymbol{X}_i, \boldsymbol{X}_j)=\exp(-\gamma\|\boldsymbol{X}_i-\boldsymbol{X}_j\|)^2$;
Sigmoid kernel: $K(\boldsymbol{X}_i, \boldsymbol{X}_j)=\tanh(C_1(\boldsymbol{X}_i \cdot \boldsymbol{X}_j)+C_2)$.

Eq. (2) shows that only penalty parameter C and the kernel function need to be ascertained to solve the regression problem using SVR, regardless of the mapping function and the concrete high dimensional feature space.

2.2 Connection between SVC and SVR

To use deductions of SVC to support our summarized rule of SVR theoretically, it is necessary to point out the connection between SVC and SVR.

The training data of SVC has the same form $(\boldsymbol{X}_i, y_i)$ as SVR, while y_i could just take the limited two values ±1 denoting two categories. As shown in Fig. 2, the regression problem to find the fitting curve $f(\boldsymbol{X})$ can be equivalently transformed to a classification problem of separating the two sets $D_+=\{(\boldsymbol{X}_i, y_i-\varepsilon), +1\}$ and $D_-=\{(\boldsymbol{X}_i, y_i+\varepsilon), -1\}$ (Bi and Bennett, 2003). Here points shifted up by ε gather around the curve $f(X)+\varepsilon$ and form D_+, and points shifted down by ε gather around the curve $f(X)-\varepsilon$ and form D_-. The only difference is that the dimensionality of the input vector $\boldsymbol{X}_i'=(\boldsymbol{X}_i, y_i+\varepsilon)$ is increased by one. Thus, we can say SVR is equivalent to SVC.

3 The preparation work

3.1 Kernel selection and order of parameter selection

As the RBF kernel is applicable in different situations for its wider convergence domain and has been accepted as the most common and ideal kernel (Scholkopf and Smola, 2002), we choose it for our experiments. Now we just need to ascertain epsilon parameter ε, penalty parameter C, and kernel width γ.

These three parameters are not entirely dependent on each other; searching for the optimal parameter one by one is therefore not advisable. The experiment results show that the estimation error map of parameter pair (C, γ) takes on a similar trend with different values of ε (details in Section 4.1), and has nearly the same optimal pair when ε varies little. This indicates that the selection of ε is independent of the pair (C, γ) to some degree (details in Section 5.1); hence, ε can be separated out. Our method to set the three optimal parameters is to set the optimal ε first and then to search for the optimal (C, γ) using the proposed method. How to set ε is illustrated in detail in Section 5.3.1. Next we focus on how to obtain the optimal pair (C, γ) first.

It is well known that letting the C and γ grow exponentially is a practical method to identify good parameters. An $r \times r$ uniform grid in the 2D logarithmic coordinate space ($C'=\log_2 C$, $\gamma'=\log_2 \gamma$) is usually used. The point in the grid represents a parameter pair (C', γ'). Here we perform experiments on a 21×21 uniform grid where C' and γ' both have a range 2^{-10}, 2^{-9}, …, 2^9, 2^{10}. Among several performance measures, we use the fivefold cross validation error (CV-error) to measure the performance (Duan *et al*., 2003).

3.2 Asymptotic behavior of (C', γ') for SVC and the linear search method

Before presenting our proposed method, we have to introduce Keerthi and Lin (2003)'s asymptotic behavior and their linear search method, since part of their conclusions can support our summarized rule for SVR theoretically. Moreover, their method will be used for a comparison with our method.

Keerthi and Lin (2003) drew two conclusions for SVC based on mathematical deduction (Fig. 3): (1) There does exist a good region (in which (C', γ') is most likely to have a small CV error), which is located in the center part of the grid and separated from the overfitting/underfitting region by a contour line. (2) When γ takes a small value (i.e., $\sigma^2=1/\gamma$ is large), all classifiers trained by points on the straight line $\log \sigma^2=\log C-\log C_0$ are nearly the same (in the logarithmic coordinate space, the line is $-\gamma'=C'-C_0'$, and its slope is −1). When γ approaches the limit value 0, the RBF SVM classifiers approximate the linear SVM classifier with parameter C_0'.

Keerthi and Lin (2003) proposed a linear search method based on the second conclusion, which first searches for the optimal $C_0'^*$ using a linear kernel and then searches for the best (C', γ') satisfying $-\gamma'=C'-C_0'$ using the RBF kernel. This method performs efficiently because it needs only to search 21+21=42 points, while the grid search method needs to search 21×21=441 points.

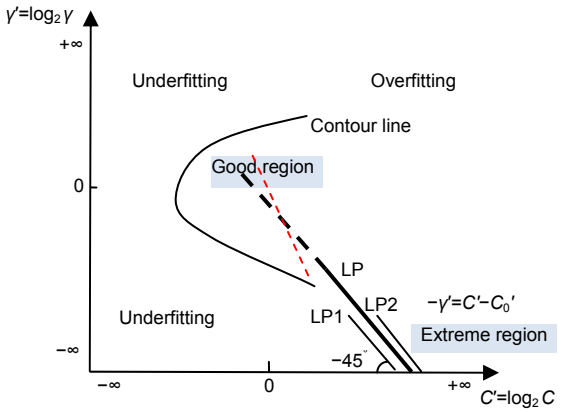

Fig. 3 Good region and straight line

The contour line does exist which seperates the good region from the underfitting/overfitting regions. In the extreme region where $\gamma'\to-\infty$, three parallel lines are satisfying the expression $-\gamma'=C'-C_0'$ with different values of C_0', and the bold line LP has the optimal $C_0'^*$

3.3 Query about the search field of linear search

There is a concern about the linear search method in theory. According to the second conclusion, we can indeed deduce the distribution rule of good points in the extreme region (where $\gamma'\to-\infty$ or $\gamma\to0$): the classifier defined by the bold line LP in Fig. 3 with good $C_0'^*$ which is searched with a linear kernel has a smaller CV error than the one defined by the line LP1 or LP2 with worse C_0'. On the other hand, there is no proof of the connection between the normal good region and the extreme region, and we cannot conclude that the classifier defined by the extension line of LP in the good region (the bold dashed line in Fig. 3) has the smallest error. Stated another way, the distribution of good points in the good region may not be the same line (bold dashed line) as the one in the extreme region (line LP). It could be another line (for example, the thin dashed line in Fig. 3) or even a triangle or quadrilateral. Thus, the linear search method may locate the wrong search field, searching for the best point in the good region directly along the extension from the best line with a fixed unit slope in the extreme region. This concern will be confirmed by the following experiment results.

4 The proposed method

4.1 Rough line rule

Extensive experiments based on simulation data and practical data were used to summarize how the good points with small CV errors are distributed. Grid maps, whose color shows the performance of SVR using the parameter point of the gird, are needed.

We simulated five sets of data using sinc function $y=a\sin x/x$ where a takes −5, −1, −0.1, 0.1, 1, 5 and another five sets by adding Gaussian noise with 15 dB to the former five sets. Eight sets of practical data with much more random noise are from UCI, Statlog, StatLib, and other collections (which are available on Lin Chih-Jen's homepage, http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/). These eight sets have different numbers of attributes and different ranges (Table 1). The number of training/test examples for all the SVR data sets is 250.

Table 1 Description of SVR data sets

SVR data set	Number of attributes	Range of the regression value
abalone	8	1–26
cpusmall	12	0–99
housing	13	12–50
mg	6	0.4–1.3
mpg	7	9–46
space_ga	6	0.1–1.0
cadata	8	6E3–5E6
bodyfat	14	0.99–1.10

We implemented experiments directly using the grid search tool in Python language which is provided in Package LibSVM 3.1 by Lin Chih-Jen. It can be used to search for the best combination of three parameters so that we can obtain several grid maps at different values of ε (i.e., 2^{-10}, 2^{-9}, …, 2^{9}, 2^{10}). For every set of training data, we can obtain 21 grid maps with 21 different values of ε. A new grid map could be obtained when the number of training data sets or the fold of the CV error varies. Due to the space limitation, we just present some typical gird maps in Fig. 4.

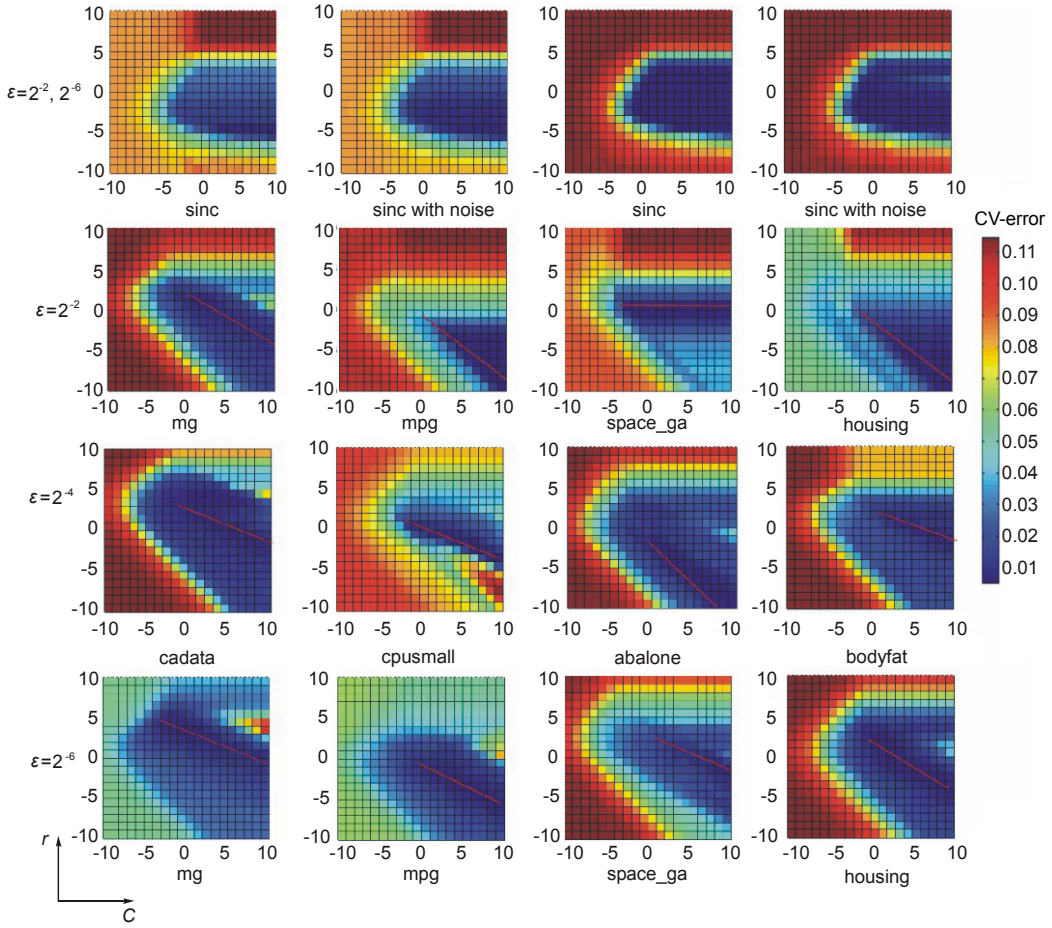

Fig. 4 Grid maps obtained using three-dimensional grid search

Coordinates of the grid map are as shown in the lower-left corner. The first row: grid maps of two sets of 200 simulation data (a=1), the left two maps for $\varepsilon=2^{-2}$ and the right two for $\varepsilon=2^{-6}$. The lower three rows: grid maps of eight sets of 250 practial data for different ε. Values of ε are marked on the left side, and names of practical data sets are marked on the bottom of each map

For the simulation data, there are smaller differences when ε or a takes a different value. In addition, grid maps always take on a similar distribution though the amount of training data varies. Hence, we just present four maps here (Fig. 4). Nothing new is illustrated, except that a good region does exist. The good region is located at the center part of the grid, which supports the first conclusion in Keerthi and Lin (2003).

As for the practical data, we also just present some typical grid maps for different sets and different ε (Fig. 4) due to their similar trends and limited space. To make it clearer, we mark out the good region by dashed lines and the rough LP line by solid lines in Fig. 5. In Fig. 4, the LP line in most maps is a rough line (used for the proposed method) with an unfixed

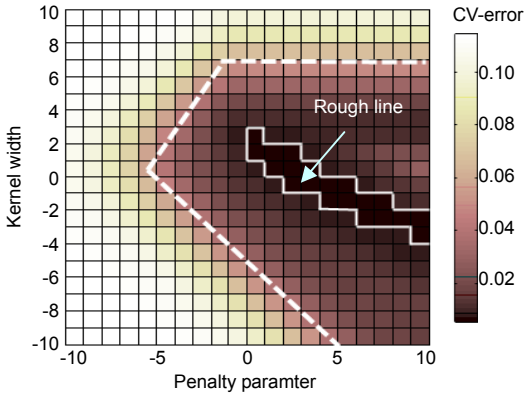

Fig. 5 Rough line cutting through the good region (housing data, $\varepsilon=2^{-6}$)

The good region is marked out by dashed lines and the rough LP line by solid lines

slope, which has a small CV error cutting through the good region, rather than a straight line with a unit slope (used for the linear search method). The rough line always radiates from the center part to a point on the below or right adjacent part. In addition, the rough line is several lattices wide. Therefore, we propose a method which searches directly along this rough line for the optimal point rather than in the whole grid or along the straight line with a fixed unit slope.

4.2 Novel linear search method

Novel linear search based on the above rough line rule first needs to ascertain the slope of the rough line by finding two points on the line. The detailed procedures are presented in the following (Fig. 6).

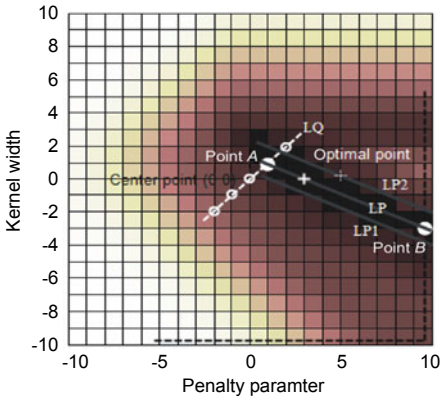

Fig. 6 Skech map of the proposed method

1. Find the first point A on the rough line (marked by larger solid dots): as the rough line always radiates from the center part, the line LQ with a unit slope passing the center point (0, 0) must intercross with the rough line LP. The intercross point of line LP and line LQ must have the smaller CV-error on line LQ. Thus, among the points next to the center point on line LQ, if point A has the smallest error, it must be on the rough line. Hence, we search five points on line LQ (marked by hollow dots) for point A, which has the smallest CV error.

2. Find the second point B on the line (marked by larger solid dots): since the rough line radiates to the bottom/right adjacent line, we search 30 points on the two adjacent lines (marked by the black dashed line) for point B, which has the smallest CV error.

3. Search for the optimal point (marked by solid thick '+') along the rough line LP: first calculate the slope of line LP determined by points A and B, and then search the points on line LP for the best point (C'^{*}, γ'^{*}).

This method needs only to search about 5+15×2+10=45 points, which are much fewer than the 441 points used in the prime grid search method. It is competitive with the linear search method, with a higher accuracy while costing nearly the same time. We name this method 'novel linear search I'.

However, the rough line is several lattices long and sometimes the optimal point may not be on line LP, but just next to it (for example, the point marked by thin dashed '+'). In this case, one widens the search field from line LP to both sides for high accuracy. This is to search along LP, LP1, and LP2 (which are shifted up or down by one unit from LP). We name this 'novel linear search II'. This method needs to search 5+15×2+10×3=65 points. As a matter of fact, novel linear search I is able to achieve nearly the same accuracy as novel linear search II in most cases. This can be seen in Section 5.1.

For a map with a rough line, this method is quite targeted while the linear search method using the straight line with a unit slope will fail because of its incorrect search field. If the map does not have a rough line with a small error, this method is still effective. For one thing, the method to find the rough line guarantees that the rough line must cut through the good region; for another, the points in the good region perform more or less the same (such as the third or fourth map of the first row in Fig. 4). Hence, this method narrows the search field from the good region to a line in the good region. This saves time while achieving nearly the same accuracy as the grid search.

5 Comparison results

5.1 Comparison of the two proposed methods

Here we compare the two methods based on eight practical SVR data sets in the case ε=0.01. Table 2 lists the optimal point, computing time, and CV error using the two methods.

Table 2 shows that, in most cases, novel linear search I can find the same optimal point as novel linear search II, but saves much more time. For the three exceptions (space_ga, cadata, and bodyfat), errors of method I increase by (0.133−0.0131)/0.0131=1.5%, (2128.2−2016.1)/2016.1=5.56%, and (1.2072−1.1994)/1.1994=0.65%, all less than 6%. Hence, novel

linear search I is more competitive and is recommended. Since SVC and SVR are equivalent, the conclusion is also established for SVC. Method I indeed performs well in all the subsequent experiments.

Table 2 Comparison of the two rough line search methods*

Data set	(C'^*, γ'^*)		CV-error		Time (s)	
	I	II	I	II	I	II
abalone	(1, −1)	(1, −1)	0.1939	0.1939	9.20	13.40
cpusmall	(5, −2)	(5, −2)	0.2475	0.2475	9.22	14.60
housing	(2, −1)	(2, −1)	0.2099	0.2099	7.30	11.80
mg	(1, 3)	(1, 3)	0.0075	0.0075	4.51	8.27
mpg	(−1, 1)	(−1, 1)	0.1910	0.1910	5.73	9.64
space_ga	(10, −8)	(10, −9)	0.0133	0.0131	5.64	9.33
cadata	(1, 1)	(2, 1)	2128.2	2016.1	6.50	10.60
bodyfat	(10, −8)	(9, −8)	1.2072E-4	1.1994E-4	3.26	5.42

* ε=0.01. I and II represent novel linear search I and novel linear search II, respectively

5.2 Comparison of three methods for SVC

To explicitly demonstrate the advantages of our method, we replicated the study of Keerthi and Lin (2003). We adopted nearly the same data sets and the same number of training/test examples (http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm). The only two exceptions are Svmguide1 and German (Table 3), because we want to make the number of attributes range evenly. Another difference is that we normalized the attributes first to improve accuracy. The genetic algorithm is included for comparison. The selection, crossover, and mutation function are stochastic uniform function, scattered crossover, and Gaussian mutation, respectively. Since the two parameters to be optimized range from −10 to 10, we set the size of the population to 20 and the generation to 100. Once fitness varies less than 1E-6, the genetic optimization will stop. Comparison results are shown in Table 4.

Table 3 Description of SVC data sets

Data set	k	N_{tr}	N_{te}	$N_=$
Banana	2	400	4900	1200
Svmguide1	4	800	300	800
Diabetes	8	168	300	400
Image	18	1300	1010	<200
German	24	500	200	
Splice	60	1000	2175	
Adult	123	1605	29589	
Web	300	2477	38994	

k is the number of attributes. N_{tr}, N_{te}, and $N_=$ are the numbers of training examples, test examples, and training examples when novel linear search and linear search cost the same time, respectively

It is shown that our method and the linear search both save considerable time compared to the prime grid search and the genetic algorithm, and the genetic algorithm is the most time-consuming. As for accuracy, the decreased accuracy (DA) is used as a benchmark, which is the relative increase in error:

$$\mathrm{DA}=\frac{\mathrm{Error}_{\mathrm{comp}}-\mathrm{Error}_{\mathrm{gs}}}{\mathrm{Error}_{\mathrm{gs}}}\times 100\%, \tag{4}$$

where $\mathrm{Error}_{\mathrm{comp}}$ is the error of the compared method and $\mathrm{Error}_{\mathrm{gs}}$ is the error of the gird search.

Fig. 7a shows DA of eight data sets using three methods compared with the prime grid search. We find that the CV-error using the novel linear search has increased by less than 7% for all cases, while the CV-error using linear search increases by more than 15% for four cases. As for the genetic algorithm, it always guarantees acceptable accuracy, but its DA curve fluctuates drastically. The genetic algorithm sometimes performs better than the grid search because its search pace is much smaller, and sometimes worse.

Table 4 Comparison of four methods for SVC

Data set	(C'^*, γ'^*)				CV-error				Time (s)			
	PG	LS	I	GS	PG	LS	I	GS	PG	LS	I	GS
Banana	(9, 4)	(−2, 5)	(8, 4)	(8.67, 4.48)	0.0725	0.0980	0.0772	0.0800	29.02	2.86	3.33	94.95
Svmguide1	(2, 2)	(5, 1)	(2, 2)	(3.90, 0.74)	0.0337	0.0385	0.0337	0.0287	89.33	7.67	6.41	74.95
Diabetes	(3, −3)	(6, −5)	(3, −3)	(0.14, −1.10)	0.2351	0.2351	0.2351	0.2500	64.26	5.49	7.59	14.17
Image	(8, 0)	(4, 3)	(5, 2)	(7.52, 1.86)	0.0278	0.0285	0.0283	0.0260	660.5	79.3	49.36	666.40
German	(6, −7)	(8, −8)	(5, −6)	(7.85, −8.17)	0.2340	0.2660	0.2340	0.2360	103.3	59.9	12.40	173.87
Splice	(2, −3)	(1, −3)	(1, −2)	(7.33, −4.48)	0.1220	0.1260	0.1280	0.1250	1319	1113	181	2817.24
Adult	(1, −4)	(1, −4)	(1, −4)	(2.80, −5.23)	0.1623	0.1623	0.1623	0.1657	4303	2851	2253	3885.5
Web	(4, −4)	(10, −9)	(4, −4)	(9.22, −9.24)	0.0171	0.0202	0.0171	0.0182	4395	2648	2451	4248.6

PG: prime grid search; LS: linear search; I: novel linear search I; GS: genetic search

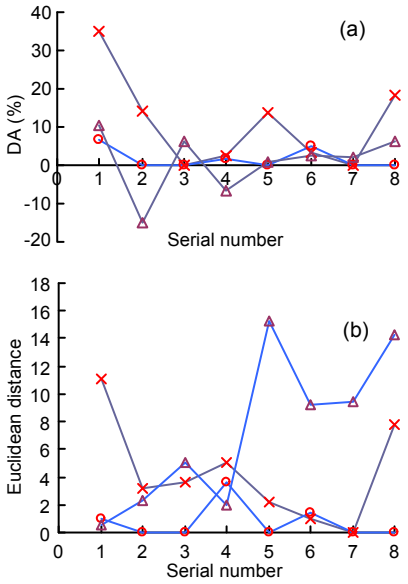

Fig. 7 Comparison of the three methods

(a) The decreased accuracy (DA) compared with grid search; (b) Euclidean distance of the two optimal points obtained using linear search, rough linear search, and genetic search compared with grid search. '○', 'Δ', and '×' represent novel linear search, linear search, and the genetic algorithm, respectively

The initial population has a great influence on the final optimal result, because it is quite easy to fall into a local optimum. For example, the Banana set had four different results when the software ran four times. Table 5 lists another three results apart from the one result in Table 4.

Table 5 Different results of the Banana set using the genetic algorithm

(C'^*, γ'^*)	CV-error
(6.96, 4.66)	0.0750
(−1.59, 5.78)	0.0875
(7.48, 4.59)	0.0700
(8.67, 4.48)	0.0800

Therefore, the genetic algorithm performs in an unstable fashion. In contrast, the novel linear search and linear search perform in a relatively stable fashion. Fig. 7a shows that the novel linear search has higher accuracy than the linear search. Fig. 7b shows that the optimal point using our method continues to be located next to the correct one using the grid search. Moreover, our method always finds the same point as the correct one using the grid search method. Hence, our method guarantees nearly the same accuracy as the grid search and performs stably. Undoubtedly, locating the correct search area is the key step for the next search for the optimal point. To further illustrate the stability of our method, we list the numbers of data sets corresponding to three different high accuracy results (ED=0, ED<3, and DA<7%) in Table 6. Euclidean distance (ED) between the optimal point searched by the compared method and the optimal one searched by the prime grid search is used to measure the accuracy of the compared method. If ED approximates 0, the compared method can guarantee nearly the same accuracy as the grid search.

Table 6 Statistics of three cases for the two methods

	Number of data sets					
	Same point*		Adjacent point**		Similar accuracy***	
	LS	I	LS	I	LS	I
SVC	1	4	2	7	4	8
SVR						
ε=0.01	1	5	2	8	1	8
ε=0.02	0	2	0	7	2	8
Optimal ε	0	3	1	7	1	8

* ED=0; ** ED<3; *** DA<7%. ED: Euclidean distance; DA: decreased accuracy. LS: linear search; I: novel linear search I

Table 6 shows that all the eight sets using the novel linear search I keep similar accuracy (DA<7%) to the grid search, indicating that our method locates the correct search area, while the optimal point found using the linear search is far from the right one, meaning that the linear search fails to locate the correct search area.

Undoubtedly, computing time costs are produced mainly during the training. For example, the computing time for the Image data set increases by less than 0.2 s and 0.4 s for the novel linear search and linear search, respectively, when the number of test examples increases from 200 to 2000. Table 4 shows that the novel linear search is much faster than the linear search when k is not smaller than 18. In addition, even if k is small, the novel linear search is still faster when the number of training examples is large (e.g., in Svmguide1). The main reason is that the linear search changes the kernel type from the linear kernel to the RBF kernel, while novel linear search continues using the RBF kernel. As we know, the SVM program needs to calculate the matrix $\boldsymbol{Q}$=

$\sum y_i y_j K(\boldsymbol{X}_i, \boldsymbol{X}_j)$ for every parameter point (C, γ). Yet, $\boldsymbol{Q}$ will not be recalculated for the same index and kernel. C decides the upper limit of α_i and further influences the number of support vectors. The support vectors alternate little when C varies. Both of the methods search for parameter points consecutively. If γ varies in the novel linear search program, only the coefficient of the RBF kernel varies and $\boldsymbol{Q}$ just needs to multiply $\exp(\gamma_{\text{present}}/\gamma_{\text{before}})$. As a result, the novel linear search just needs to calculate the alternating support vectors for the new parameter. The linear search needs, however, to recalculate $\boldsymbol{Q}$ when the kernel changes, which will cost more time if there are more training examples or attributes. The last column in Table 3 provides a simple proof for this. Apart from kernel type, the numbers of training examples and attributes are other factors. $N_=$ is the number of training examples when the novel linear search and the linear search have an equal cost. If the number of training examples is larger than $N_=$, the novel linear search method will cost less time. $N_=$ does not exist for the last four data sets, because small training examples cannot provide enough information to obtain an accurate SVM model. That is, there are too many attributes and the model is complex. Hence, the novel linear search will cost less time if k is larger than 24.

Therefore, a conclusion can be drawn that, our method not only locates the correct search field and guarantees nearly the same accuracy as the grid search (which is much more accurate than the linear search), but also saves much more time compared to the grid search and costs more or less the same time as the linear search when the number of training examples or attributes is small. However, if the number of training examples or attributes is large, our method has overwhelming time superiority to the linear search. The results show that the genetic algorithm can guarantee an acceptable accuracy in nearly every case, and sometimes it performs even better than the grid search and our method (because the search pace of the grid search is large). However, the genetic algorithm consumes more time and performs unstably, more easily falling into a local optimum, and has a different result for a new running. Thus, the genetic algorithm greatly relies on the choice of its own initial parameters. A deeper understanding of the genetic algorithm may solve these problems (Gijsberts *et al*., 2010).

5.3 Comparison of three methods for SVR

Next we will illustrate that the above conclusion is also established for SVR. We conducted similar experiments and made the same analysis as for SVC. The only difference is that we have to set another parameter ε.

5.3.1 Setting of ε

The optimal (C'^*, γ'^*) is different when ε varies, because ε decides the number of support vectors; that is, a smaller ε results in more support vectors and consequently a more accurate SVM model (Deng and Tian, 2004). And the variation of ε will greatly increase the search time because y_i varies and to calculate $\boldsymbol{Q}=\sum y_i y_j K(\boldsymbol{X}_i, \boldsymbol{X}_j)$ requires recalculating the alternating support vectors. Therefore, we need to set ε first and then search for the optimal (C'^*, γ'^*). Table 7 shows how an optimal ε helps to obtain high accuracy. Here we use Eq. (5) (Cherkassky and Ma, 2004) to set ε directly, which provides better performance in terms of prediction risk and robustness:

$$\varepsilon_0=\tau\sigma\sqrt{\frac{\ln n}{n}},\ \sigma=\frac{1.5}{n}\sum_{i=1}^{n}(y_i-\bar{y})^2,\ \tau=3. \quad (5)$$

However, in Fig. 4 we find that grid maps in the last row with smaller ε always have more obvious features of a rough line in the good region. Thus, to guarantee the effectiveness of the rough line method and to achieve a better performance, we should make sure that the optimal ε_0 calculated using Eq. (5) is not larger than the upper limit 0.2. An extremely small ε will increase the search time dramatically and even make it impossible to obtain the SVM model, because two convex hulls are not separable now (Bi and Bennett, 2003). Thus, the lower limit is necessary, and we use 0.001. The limits are chosen according to results of our experiments. If one wants to skip the setting of ε, just set it to 0.001, which can guarantee an acceptable accuracy. In this work, ε is set as follows:

$$\varepsilon=\begin{cases}0.001, & \varepsilon_0\le 0.001,\\ \varepsilon_0, & 0.001<\varepsilon_0<0.2,\\ 0.2, & \varepsilon_0\ge 0.2.\end{cases} \quad (6)$$

Table 7 lists the results of eight SVR data sets using the above method to search for the optimal ε, which will be further analyzed by comparing results with different ε.

Table 7 The optimal ε of eight SVR data sets

Data set	k	Optimal ε	Data set	k	Optimal ε
abalone	8	0.20	mpg	7	0.05
cpusmall	12	0.015	space_ga	6	0.03
housing	13	0.01	cadata	8	0.03
mg	6	0.04	bodyfat	14	0.001

5.3.2 Comparison results for SVR

Here we present the comparison results of three methods based on eight practical SVR data sets when ε takes 0.01 and 0.02. Then we use the above method to set optimal ε and include the genetic algorithm in the comparison.

Tables 8c and 9 show that the genetic algorithm for SVR also conforms to the conclusion for SVC. Then we still focus on the analysis of the other two methods. Table 8 directly shows that our method shortens the search time by more than a factor of two

Table 8 Comparison of searching methods with different ε for SVR*

(a) ε=0.01

Data set	(C'^*, γ'^*)			CV-error			Time (s)		
	PG	LS	I	PG	LS	I	PG	LS	I
abalone	(1, −1)	(10, −1)	(1, −1)	0.1939	0.2218	0.1939	37.8	8.65	9.20
cpusmall	(5, −2)	(1, −1)	(5, −2)	0.2475	0.2692	0.2475	34.6	9.10	9.22
housing	(2, −1)	(9, 1)	(2, −1)	0.2099	0.3079	0.2099	33.2	7.12	7.30
mg	(1, 3)	(−4, 1)	(1, 3)	0.0075	0.0087	0.0075	28.7	3.29	4.51
mpg	(−1, 1)	(−1, 2)	(−1, 1)	0.1910	0.1956	0.1910	19.2	4.50	5.73
space_ga	(10, −9)	(2, 8)	(10, −8)	0.0131	0.0156	0.0133	30.7	4.50	5.64
cadata	(2, 1)	(−3, 0)	(1, 1)	2016.1	2272.7	2128.2	45.6	5.32	6.50
bodyfat	(9, −8)	(10, 0)	(10, −8)	1.1994E-4	3.4119E-4	1.2072E-4	9.2	3.48	3.26

(b) ε=0.02

Data set	(C'^*, γ'^*)			CV-error			Time (s)		
	PG	LS	I	PG	LS	I	PG	LS	I
abalone	(0, −1)	(4, 6)	(0, −1)	0.1928	0.2202	0.1928	26.56	5.31	5.70
cpusmall	(6, −2)	(1, 0)	(6, −2)	0.2419	0.2573	0.2419	22.48	5.61	6.51
housing	(2, −2)	(9, 1)	(2, −1)	0.2100	0.3340	0.2214	19.02	4.39	4.19
mg	(1, 4)	(−4, 2)	(1, 3)	0.0069	0.0084	0.0070	25.50	2.63	2.20
mpg	(0, 0)	(7, −4)	(0, 0)	0.1869	0.1933	0.1869	19.37	3.90	4.52
space_ga	(10, −9)	(2, 7)	(10, −8)	0.0127	0.0157	0.0128	22.80	3.82	4.54
cadata	(2, 1)	(−2, −1)	(0, 2)	2016.1	2283.7	2019.4	32.60	4.13	4.70
bodyfat	(8, −7)	(8, 1)	(10, −7)	1.5616E-4	3.9291E-4	1.6174E-4	4.00	1.40	0.86

(c) Optimal ε

Data set	(C'^*, γ'^*)				CV-error				Time (s)			
	PG	LS	I	GS	PG	LS	I	GS	PG	LS	I	GS
abalone	(1, 0)	(10, 0)	(1, 0)	(0.91, 0.05)	0.1757	0.1758	0.1757	0.1755	1.15	0.23	0.29	3.82
cpusmall	(6, −2)	(0, 0)	(6, −2)	(−0.13, −0.25)	0.2417	0.2650	0.2417	0.2560	28.42	6.78	6.47	32.85
housing	(2, −2)	(9, 1)	(2, −1)	(−1.88, −2.78)	0.2099	0.3079	0.2099	0.3989	33.20	7.12	7.30	47.45
mg	(1, 4)	(−4, 2)	(3, 2)	(1.83, 2.98)	0.0069	0.0083	0.0075	0.0070	13.70	1.79	1.85	40.20
mpg	(3, 0)	(−2, 0)	(3, 0)	(3.21, −2.40)	0.1806	0.2173	0.1806	0.2377	6.50	1.95	1.95	12.15
space_ga	(10, −9)	(2, 7)	(10, −8)	(0.52, 0.26)	0.0127	0.0157	0.0128	0.0150	22.80	3.82	4.54	24.31
cadata	(1, 2)	(3, −4)	(0, 2)	(0.83, 1.75)	2015.1	2267.3	2018.5	1999.6	27.90	3.79	4.20	33.04
bodyfat	(10, −8)	(9, 0)	(10, −7)	(3.49, −1.05)	1.0015E-4	2.9927E-4	1.0051E-4	0.9527E-4	19.77	6.36	6.50	128.05

* The numbers of training/test data are both 250. PG: prime grid search; LS: linear search; I: novel linear search I; GS: genetic search

compared with the grid search, and uses almost the same time as the rough line search. Fig. 8a shows that the CV-error using our method increases by less than 6% for all cases, while the CV-error using linear search increases by more than 22% for most cases and even more than double for some cases (housing and cadata), which means these searches fail. The statistical results are shown in Table 6.

Table 9 Different results of the cpusmall set using the genetic algorithm

(C'^*, γ'^*)	CV-error
(1.31, −0.54)	0.2499
(0.77, 0.94)	0.2258
(0.34, −0.12)	0.2660

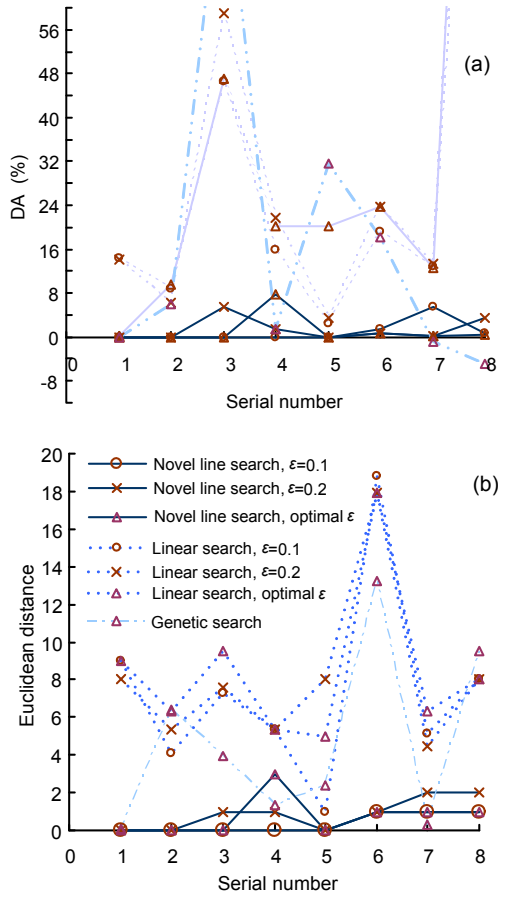

Fig. 8 Comparison of linear search and rough line search
(a) The decreased accuracy (DA) compared with grid search; (b) Euclidean distance of the two optimal points obtained using linear search and rough linear search

Comparing three cases for every SVR data set, we find that the CV-error of every SVR data set using the optimal ε is smaller than those in the two cases using ε=0.01 or ε=0.02 (Table 8). This implies that our method of setting ε is effective and feasible.

Putting the above together, both theoretically and empirically, we conclude that our method which locates the correct search field has overwhelming advantages of accuracy and stability with competitive computing time. Compared with the genetic algorithm, our method has three advantages: (1) it has good stability and hunts the global optimum, not a local optimum; (2) it is simple to use because it does not need to set parameters; (3) it has overwhelming time superiority when the number of attributes or training examples is large.

6 Conclusions

We summarize the rough line rule of SVR parameters (C, γ) based on extensive experiments and propose a novel linear search method, which proves competitive in terms of efficiency, accuracy, and stability for both SVC and SVR, both theoretically and empirically. As for the epsilon parameter, a direct-setting formula with thresholds also proves feasible and effective. The example of motion prediction indicates that SVR using our method is feasible in setting the epsilon parameter ε, penalty parameter C, and kernel width γ in a real application.

While there is still a large space for improving the operational efficiency, we can try to shrink the sample set strategically or use the optimization algorithms for SVM internal training, such as the SVM-light algorithm and sequential minimal optimization. The robustness of SVM for noise is also the next focus of our research. A new geometry-based (GB) criterion is shown to be more competitive and stable than the CV-error (Ahn, 2010), and may help to maintain the nice properties of the rough line rule and enhance the robustness of our method. Thus, the GB criterion is a good choice for robustness.

References

Ahn, J., 2010. A stable hyperparameter selection for the Gaussian RBF kernel for discrimination. *Statist. Anal. Data Min.*, **3**(3):142-148. [doi:10.1002/sam.10073]

Bengio, Y., 2000. Gradient-based optimization of hyperparameters. *Neur. Comput.*, **12**(8):1889-1900. [doi:10.1162/089976600300015187]

Bi, J., Bennett, K.P., 2003. A geometric approach to support vector regression. *Neurocomputing*, **55**(1-2):79-108.

[doi:10.1016/S0925-2312(03)00380-1]

Chapelle, O., Vapnik, V., Bousquet, O., Mukherjee, S., 2002. Choosing multiple parameters for support vector machines. *Mach. Learn.*, **46**(1/3):131-159. [doi:10.1023/A:1012450327387]

Cherkassky, V., Ma, Y.Q., 2004. Practical selection of SVM parameters and noise estimation for SVM regression. *Neur. Networks*, **17**(1):113-126. [doi:10.1016/S0893-6080(03)00169-2]

Cristianini, N., Kandola, J., Elisseeff, A., Shawe-Taylor, J., 2006. On kernel target alignment. *Innov. Mach. Learn.*, **194**:205-256. [doi:10.1007/3-540-33486-6_8]

Deng, N.Y., Tian, Y.G., 2004. A New Method in Data Mining—Support Vector Machine. Science Press, Beijing, China (in Chinese).

Duan, K., Keerthi, S.S., Poo, A.N., 2003. Evaluation of simple performance measures for tuning SVM hyperparameters. *Neurocomputing*, **51**:41-59. [doi:10.1016/S0925-2312(02)00601-X]

Gijsberts, A., Metta, G., Rothkrantz, L., 2010. Evolutionary optimization of least-squares support vector machines. *Data Min.*, **8**(4):277-297. [doi:10.1007/978-1-4419-1280-0_12]

Huang, C.L., Dun, J.F., 2008. A distributed PSO-SVM hybrid system with feature selection and parameter optimization. *Appl. Soft Comput.*, **8**(4):1381-1391. [doi:10.1016/j.asoc.2007.10.007]

Huang, C.L., Wang, C.J., 2006. A GA-based feature selection and parameters optimization for support vector machines. *Expert Syst. Appl.*, **31**(2):231-240. [doi:10.1016/j.eswa.2005.09.024]

Keerthi, S.S., Lin, C.J., 2003. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neur. Comput.*, **15**(7):1667-1689. [doi:10.1162/089976603321891855]

Kim, D., Song, J., Lee, J., Choi, B., 2007. Support vector machine learning for region-based image retrieval with relevance feedback. *ETRI J.*, **29**(5):700-702. [doi:10.4218/etrij.07.0207.0037]

Lau, K.W., Wu, Q.H., 2008. Local prediction of non-linear time series using support vector regression. *Pattern Recogn.*, **41**(5):1539-1547. [doi:10.1016/j.patcog.2007.08.013]

Po, L.M., Ma, W.C., 2002. A novel four-step search algorithm for fast block motion estimation. *IEEE Trans. Circ. Syst. Video Technol.*, **6**(3):313-317. [doi:10.1109/76.499840]

Roohi, M., Mirjalily, G., Sadeghi, M.T., 2007. Face Detection Using a Modified SVM-Based Classifier. 7th Int. Conf. on Computational Intelligence and Multimedia Applications, p.356-360. [doi:10.1109/ICCIMA.2007.243]

Sapankevych, N., Sankar, R., 2009. Time series prediction using support vector machines, a survey. *IEEE Computat. Intell. Mag.*, **4**(2):24-38. [doi:10.1109/MCI.2009.932254]

Scholkopf, B., Smola, A.J., 2002. Learning with Kernels. MIT Press, Cambridge.

Suykens, J.A.K., Vandewalle, J., de Moor, B., 2001. Optimal control by least squares support vector machines. *Neur. Networks*, **14**(1):23-35. [doi:10.1016/S0893-6080(00)00077-0]

Vapnik, V., Golowich, S.E., Smola, A., 1997. Support Vector Method for Function Approximation, Regression Estimation, and Signal Processing. Advances in Neural Information Processing Systems 9 – Proc. Conf. on Advances in Neural Information Processing Systems, p.281-287.

Vladimir, N., Vapnik, V., 2000. The Nature of Statistical Learning Theory. Springer Verlag, New York.

Wang, T.Y., Chiang, H.M., 2007. Fuzzy support vector machine for multi-class text categorization. *Inform. Process. Manag.*, **43**(4):914-929. [doi:10.1016/j.ipm.2006.09.011]

Zhu, S., Ma, K.K., 1997. A New Diamond Search Algorithm for Fast Block Matching Motion Estimation. Proc. Int. Conf. on Information, Communications and Signal Processing, **1**:292-296. [doi:10.1109/ICICS.1997.647106]