Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 8 (19): 3407-3414, 2008
ISSN 1812-5654

DSKR: A Direct Sparse Kernel Regression

[1]Chen Xiao-Feng, [1,2]Wang Shi-Tong and [2]Cao Su-Qun
[1]School of Information, Jiangnan University, Wuxi 214122, China
[2]Department of Mechanical Engineering, Huaiyin Institute of Technology, Huaian 223001, China

Abstract: The number of support vectors plays a crucial role in designing fast kernel machine. The prediction time is linearly depend on the size of training example set and smaller set of support vectors leads to faster classification or regression. When applications require high prediction speed, SVM may not perform well. To overcoming this problem, recently, a direct sparse kernel learning framework was proposed to deal with binary classification problem, which can reduce the size of support vector set efficiently. In this research, we extend the direct sparse kernel learning framework to regression problem and present a method named DSKR to build sparse kernel regression. The DSKR has two advantages as follows: (1) it can obtain sparse kernel regression with arbitrary user-defined number of support vectors; (2) it achieves less regression error than traditional SVR with less number of support vectors. Experimental comparisons are made with other kernel regression methods on both synthetic and real-world data sets. The comparisons show that the proposed DSKR gives promising results.

Key words: Sparse kernel regression, kernel learning, sparse learning, expansion vector, support vector machine

INTRODUCTION

Kernel methods have been used widely in many aspects of pattern recognition and machine learning community, such as classification, regression and clustering (Vapnik, 1998; Ben *et al*., 2001). In kernel methods, examples in input space are mapped to feature space by utilizing kernel functions. Kernel methods are usually solved by many optimization methods. Traditionally, examples in training set with non-zero Lagrangian coefficiences are support vectors, which can be used to construct kernel machines. The number of support vectors plays a crucial role in designing fast kernel machine. Steinwart (2003) suggested that the number of support vectors is linearly depend on the size of training set, so cutting down the number of support vectors is a important way for faster classification or regression.

Many methods have been proposed for reducing the size of support vector set since decades ago. Here we only survey the methods which are relative to this research and presented in recent few years. Schölkopf *et al*. (2000) proposed v-support vector machine (v-SVM)) which optimizing the number of support vectors by a adjustable parameter v. The methods in Downs *et al*. (2001) and Thies and Weber (2006) can be viewed as taking a post-processing step to simplify kernel machines. The studies in Tipping (2001), Roth (2001) and Lawrence *et al*. (2002) yielded fast kernel machines from the view of probability theory. The methods in Williams and Seeger (2001), Nair *et al*. (2002) and Drineas and Mahoney (2005) reduced the training complexity through replacing original kernel matrix by its sparse version. The methods in Vincent and Bengio (2002) and Keerthi *et al*. (2006) generated sparse kernel machines by taking basic function selection methods. The reduced support vector machine (RSVM) is another fast training kernel method (Lin and Lin, 2003; Lee and Huang, 2007). RSVM selects user-defined number of random examples from training set, i.e., the subset of training set, to train kernel machines. The advantage of RSVM is its fast training property. However, the size of support vectors set of RSVM is not always small with relatively small amount of training examples. In some experimental results, RSVM even needs much more support vectors than traditional SVM (Lin and Lin, 2003). Significant vector learning for regression was proposed for nonlinear system identification (Gao and Shi, 2005; Gao *et al*., 2007). Wang *et al*. (2006) proposed a method for sparse support vector regression based on orthogonal forward selection. Direct sparse kernel learning framework is a novel framework for reducing run-time complexity which was presented in Wu *et al*. (2006).

Corresponding Author: Wang Shi-Tong, School of Information, Jiangnan University, Wuxi 214122, China
Tel: 086-0510-85912151

In this research, we follow these studies and consider the problem of building direct sparse kernel regression through optimizing the size of support vector set directly.

MATERIALS AND METHODS

Direct sparse kernel learning framework: First, let us briefly review direct sparse kernel learning framework. The framework was presented in Wu *et al.* (2006). The general form of the direct sparse kernel learning framework can be written as follows:

$$\underset{w,b,\xi,\beta,Z}{\text{minimize}} \quad f(w,b,\xi), \tag{1}$$

$$\text{Subject to} \quad g_i(w,b,\xi) \le 0, \quad 1 \le i \le N_g \tag{2}$$

$$h_j(w,b,\xi) \le 0, \quad 1 \le j \le N_h \tag{3}$$

$$w = \sum_{i=1}^{N_z} \Phi(z_i)\beta_i \tag{4}$$

where, $f(w,b,\xi)$ is the object function. W and b are, respectively weigh vector and bias. $\xi = [\xi_1,...,\xi_{N_\xi}]^T$ is a vector of slack variables. N_ξ is the number of slack variables. N_g and N_h are the number of inequality constraints and the number of equality constraints, respectively. z_i is an expansion vector and $Z = [z_1,...z_{N_z}] \in R^{x \times N_z}$ is the matrix of expansion vector. d is the dimension of input space. N_z is the number of expansion vectors and it is a user-defined parameter. β_i is the coefficient of the expansion vector z_i and $\beta = [\beta_1,...,\beta_{N_z}]^T$ is the coefficient vector of expansion vectors. $\Phi(.)$ is a mapping function which maps examples from input space into feature space. Note that instead of support vector in traditional kernel methods, expansion vector is used in sparse kernel learning framework, since the obtained expansion vectors may not be in training set. For the sake of simplicity, we call the vectors for constructing kernel machine expansion vector uniformly without considering they are in training set or not.

Different from traditional kernel learning methods, given user-defined N_z, direct sparse kernel learning framework adds an non-convex constraint Eq. 4 into its general form. The constraint guarantees the number of expansion vectors in obtained kernel machine equals to N_z.

Since the constraint Eq. 4 is non-convex, Eq. 1-4 can not be solved with a global optimal solution. So a two-part iterative strategy is used to deal with the optimizing problem. In first part, the expansion vectors $Z = [z_1,...z_{N_z}]$ in Eq. 1-4 are fixed, by solving Eq. 1-4, we can obtain the corresponding coefficients $\beta = [\beta_1,...,\beta_{N_z}]^T$ of the fixed Z. In the second part, with fixing the obtained β and Z is optimized to minimize the object function in Eq. 1. The two parts execute iteratively until a convergent solution of Z and β is met.

Based on the framework Wu *et al.* (2006) proposed a sparse large margin method for binary classifier named Sparse Large Margin Classifier (SLMC). In this research, we extend the direct sparse kernel learning framework to regression problem and proposed Direct Sparse Kernel Regression (DSKR) method. Below let us give its detailed description.

DSKR: A method to build sparse kernel regression directly: Consider the training set $\{(x_i,y_i)\}_{i=1}^N$ with the user-defined number N_z, where $x_i \in R^d$ is the input example, y_i is the output target of x_i. N is the size of training set. To obtain a sparse kernel regression, we solve the optimizing problem as follows:

$$\underset{w,\xi,b,\beta,Z}{\text{minimize}} \quad \frac{1}{2}w^Tw + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \tag{5}$$

$$\text{Subject to} \quad y_i - w^T F(x_i) - b \le e + \xi_i \tag{6}$$

$$w^T F(x_i) + b - y_i \le e + \xi_i^* \tag{7}$$

$$\xi_i, \xi_i^* \ge 0 \tag{8}$$

$$w = \sum_{i=1}^{N_z} \beta_i \Phi(z_i) \tag{9}$$

where, C is trade-off parameter.

We solve Eq. 5-9 under the direct sparse kernel framework. First, by fixing Z, Eq. 5-9 can be transformed into the problem below:

$$\underset{w,\xi,b,\beta}{\text{minimize}} \quad \frac{1}{2}w^Tw + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \tag{10}$$

$$\text{Subject to} \quad y_i - w^T\Phi(x_i) - b \le e + \xi_i \tag{11}$$

$$w^T\Phi(x_i) + b - y_i \le e + \xi_i^* \tag{12}$$

$$\xi_i, \xi_i^* \ge 0 \tag{13}$$

$$w = \sum_{i=1}^{N_z} \beta_i \Phi(z_i) \tag{14}$$

Then we derive the dual problem of Eq. 10-14. Substituting Eq. 14 into Eq. 11 and 12, we have:

$$y_i - \beta^T \Psi_Z(x_i) - b \le \varepsilon + \xi_i \quad (15)$$

$$\beta^T \Psi_Z(x_i) + b - y_i \le \varepsilon + \xi_i^* \quad (16)$$

Where:

$$\Psi_z(x_i) = [K(z_1, x_i), ..., K(z_{N_z}, x_i)]^T \quad (17)$$

$K(z_j,x_i)$ denotes $\Phi(z_j)^T \Phi(x_i)(j = 1,...,N_z; i = 1,...,N)$.

Substituting Eq. 13-16 into Eq. 10, by introducing the Lagrangian multiplier, we have:

$$\begin{aligned} L(\xi, b, \beta, \alpha) = {} & \frac{1}{2}\beta^T K^Z \beta + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) - \\ & \sum_{i=1}^{N}\alpha_i(\varepsilon + \xi_i - y_i + \beta^T \Psi_Z(x_i) + b) - \\ & \sum_{i=1}^{N}\alpha_i^*(\varepsilon + \xi_i^* - \beta^T \Psi_Z(x_i) - b + y_i) - \\ & \sum_{i=1}^{N}\mu_i\xi_i - \sum_{i=1}^{N}\mu_i^*\xi_i^* \end{aligned} \quad (18)$$

where, $\alpha_i \ge 0$, $\alpha_i^* \ge 0$, $\mu_i \ge 0$, $\mu_i^* \ge 0 (i = 1,...,N)$ are Lagrangian multipliers, K^z is the kernel matrix of Z with $K_{ij}^z = K(z_i,z_j)$.

Setting to zero the derivative of L with respect to β, b, ξ_i and ξ_i^*, we immediately have

$$\frac{\partial L}{\partial \beta} = K^Z \beta - \sum_{i=1}^{N}\alpha_i \Psi_Z(x_i) + \sum_{i=1}^{N}\alpha_i^* \Psi_Z(x_i) = 0 \quad (19)$$

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{N}\alpha_i + \sum_{i=1}^{N}\alpha_i^* = 0 \quad (20)$$

$$\frac{\partial L}{\partial \xi_i} = C - \alpha_i - \mu_i = 0 \quad (21)$$

$$\frac{\partial L}{\partial \xi_i^*} = C - \alpha_i^* - \mu_i^* = 0 \quad (22)$$

Eq. 19 leads to

$$\beta = (K^Z)^{-1}\sum_{i=1}^{N}(\alpha_i - \alpha_i^*)\Psi_Z(x_i) \quad (23)$$

Substituting Eq. 20-23 into Eq. 18, we arrive at the dual problem of Eq. 10-14:

$$\begin{aligned} \underset{\alpha,\alpha^*}{\text{minmize}} \quad & \frac{1}{2}\sum_{i,j=1}^{N}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\hat{K}_Z(x_i,x_j) + \varepsilon\sum_{i=1}^{N}(\alpha_i + \alpha_i^*) - \\ & \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)y_i \end{aligned} \quad (24)$$

$$\text{Subject to } \sum_{i=1}^{N}(\alpha_i - \alpha_i^*) = 0 \quad (25)$$

$$\alpha_i, \alpha_i^* \in [0 \ C] \quad (26)$$

Where:

$$\hat{K}_Z(x_i,x_j) = \Psi_Z(x_i)^T (K^Z)^{-1} \Psi_Z(x_j) \quad (27)$$

In summary, we describe the proposed method DSKR as follows:

- Initialize the expansion vectors $Z = [z_1,...,z_{N_Z}]$ from the training set.
- Obtain kernel regression with two-part strategy
  - Terminate if a convergent optimizing solution of both Z and β is met.
  - Fix the expansion vectors Z, solve optimizing problem Eq. 24-26 and obtain the expansion coefficiences β.
  - Fix β obtained in step 2b, search optimal Z to minimize Eq. 24, then go back to step 2a.
- Output the obtained kernel regression.

In step 1, we initialize the expansion vectors $z_i (i = 1,...,N_z)$. Two methods may be used for this aim (Wu *et al.*, 2006). They are random selection method and clustering selection method. In the first method, N_z examples are randomly chosen from the training set to serve as initial expansion vectors. While in the second method, the training set are firstly partitioned into N_z groups by clustering methods such as K-means (Wu *et al.*, 2006), then the clustering centers is used to initialize the expansion vectors.

In step 2b, Eq. 24-26 with the fixed Z is actually a quadratic programming problem which had been well solved by support vector machine (Vapnik, 1998).

In step 2c, with the fixed β, Eq. 24 becomes the following optimizing problem:

$$\begin{aligned} W(Z) = {} & \frac{1}{2}\sum_{i,j=1}^{N}(\alpha_i^Z - \alpha_i^{Z^*})(\alpha_j^Z - \alpha_j^{Z^*})\hat{K}_Z(x_i,x_j) + \\ & \varepsilon\sum_{i=1}^{N}(\alpha_i^Z + \alpha_i^{Z^*}) - \sum_{i=1}^{N}(\alpha_i^Z - \alpha_i^{Z^*})y_i \end{aligned} \quad (28)$$

Since Eq. 28 is an unconstrained minimization problem, Newton's method and its variations are the appropriate choices for solving it. Here we use limited-memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) method (Liu and Nocedal, 1989) to handle it.

As mentioned earlier, $Z = [z_1,...z_{N_Z}] \in R^{d \times N_Z}$. Let Z_{uv} denote an arbitrary element in Z, which locates in the u-th $(1 \le u \le d)$ row and the v-th $(1 \le v \le N_z)$ column of Z. In other words, Z_{uv} is the u-th attribute of the expansion vector z_v. In terms of LBFGS, $\nabla W(Z)$ can be computed as follows:

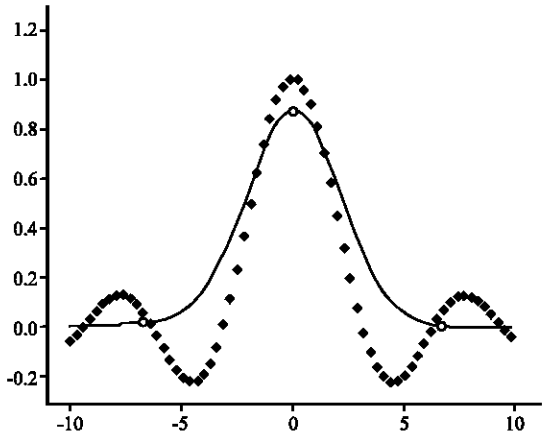

Fig. 1: Experimental result of DSKR on sinc function with N_z = 3. The small dots and circles are training examples and expansion vectors, respectively. Solid line shows the obtained regression result

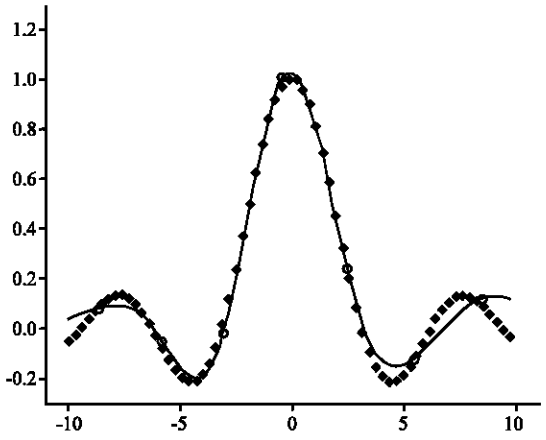

Fig. 2: Experimental result of DSKR on sinc function with N_z = 7

$$\frac{\partial W}{\partial Z_{uv}} = \frac{1}{2}\sum_{i,j=1}^{N}(\alpha_i^Z - \alpha_i^{Z^*})(\alpha_j^Z - \alpha_j^{Z^*})\frac{\partial \hat{K}_Z(x_i,x_j)}{\partial Z_{uv}} \quad (29)$$

Where:

$$\begin{aligned}\frac{\partial \hat{K}_Z(x_i,x_j)}{\partial Z_{uv}} = &(\frac{\partial \Psi_Z(x_i)}{\partial Z_{uv}})^T (K^Z)^{-1}\Psi_Z(x_j) + \\ &\Psi_Z(x_i)^T (K^Z)^{-1}(\frac{\partial \Psi_Z(x_j)}{\partial Z_{uv}}) + \\ &\Psi_Z(x_i)^T \frac{\partial (K^Z)^{-1}}{\partial Z_{uv}}\Psi_Z(x_j)\end{aligned} \quad (30)$$

$$\frac{\partial (K^Z)^{-1}}{\partial Z_{uv}} = -(K^Z)^{-1}\frac{\partial K^Z}{\partial Z_{uv}}(K^Z)^{-1} \quad (31)$$

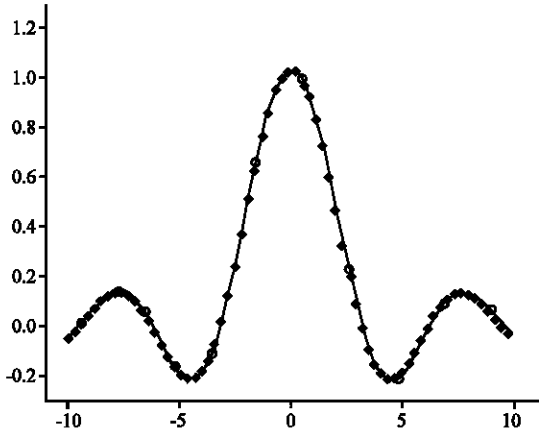

Fig. 3: Experimental result of DSKR on sinc function with N_z = 11

where, α_i^z and α_i^{z*} are the corresponding Lagrangian multiplier of the fixing x_i(i = 1,...,N).

Sinc Z_{uv} is the u attribute of the expansion vector z_v and $\Psi_z(x_i) = [K(z_1,x_i),...,K(z_{Nz},x_i)]^T$, so other expansion vectors don't contain element Z_{uv}, we have $\frac{\partial K(z_{v'},x_i)}{\partial Z_{uv}} = 0$ (V' = 1,...,v−1, v+1,...N_z) $\frac{\partial \Psi_Z(x_i)}{\partial Z_{uv}}$ in Eq. 30 can be computed as follows:

$$\begin{aligned}\frac{\partial \Psi_Z(x_i)}{\partial Z_{uv}} &= [\frac{\partial K(z_1,x_i)}{\partial Z_{uv}},...,\frac{\partial K(z_v,x_i)}{\partial Z_{uv}},...,\frac{\partial K(z_{N_z},x_i)}{\partial Z_{uv}}]^T \\ &= [0,...,\frac{\partial K(z_v,x_i)}{\partial Z_{uv}},...0]^T\end{aligned} \quad (32)$$

In the following, we illustrate the application of DSKR to regression using a toy experiment. The data set is generated by sinc function, i.e., f(x) = sin(x)/x+t, where x takes from a uniform distribution over the interval [-10 10]. t denotes a random noise generated by MATLAB code 0.001×randn (1,1). The data set contains 67 examples. In the experiments, c = 300, ε = 0.001. Gaussian kernel function $K(x_i,x_j) = \exp(-r\|x_i-x_j\|^2)$ with r = 3 is used. Expansion vectors are initialized by K-means clustering method. Figure 1-3, respectively show the experimental results with N_z = 3, 7, 11. From these results, we can see that DSKR generates sparse regression with N_z expansion vectors and N_z is relative to the regression error. With a small N_z, such as N_z = 3, the regression error takes large value. However, with the increasing of N_z, the regression error decreases gradually, when N_z = 11, the performance of DSKR is quiet good. A simple MATLAB implement of DSKR for the experiment is available at

(http://combustion.ustc.edu.cn/renwu/Xiaofeng%20Chen%27s%20Homepage.htm.).

RESULTS AND DISCUSSION

Experimental settings: Here, we study the performance of DSKR. We compare DSKR with ε-support vector regression (ε-svr), v-support vector regression (v-svr) (Schölkopf *et al*., 2000) and Relevant Vector Machine (RVM) (Tipping, 2001) on both synthetic and real-world data sets. v-svr generates sparse kernel regression by adjusting parameter v. RVM is a sparse kernel regression which is often used as a baseline to measure the performance of sparse kernel regression methods (Roth, 2001; Wu *et al*., 2006; Gao *et al*., 2007). The code of RVM is available at (http://www.miketipping. com/.) ε-svr and v-svr are implemented by LIBSVM which can be download from (http://www.csie.ntu.edu.tw/~cjlin/libsvm/.) Gaussian kernel $K(x_i,x_j) = \exp(-r\|x_i-x_j\|^2)$ is used in all experiments. The parameters in experiment are chosen as below: ε is computed by

$$\varepsilon = 0.01 \times \frac{1}{N}\sum_{i=1}^{N} y_i .$$

The optimal parameter C and r are selected from 5-fold cross validation by using ε-svr on training set (Tipping, 2001). The obtained C and r can lead to the best generalization performance. For v-svr, we adjust v to get the desired number of expansion vectors. For DSKR, we set N_z directly. We give the experimental results of DSKR with both random selection and K-means clustering based initialization. All the experiments are run on a personal computer with Intel 1.4 GHz processors, 384 M memory and Windows 2000.

Two error criteria are used for performance comparison:

- Mean Absolute Error (MAE)

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - f(x_i)| \tag{33}$$

- Mean Relative Error (MRE)

$$MRE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i - f(x_i)}{y_i}\right| \times 100\% \tag{34}$$

Data sets: Our experiments perform on seven data sets. Among them, two are synthetic while five are real-world data sets.

Two synthetic data sets are shown in Table 1 and they are sinc1 and 3-D Mexican hat1. Different noises are added into both data sets for comparing the robustness of DSKR with other three methods. Let y denotes the output targets, snr denotes the signal-to-noise ratio, noise is added by MATLAB code ynoise = awgn (y, snr, 'measured'). Three ratios snr = 10, 15 and 25 are used.

Five real-world regression data sets are bodyfat, housing, mpg, pyrim and triazines which are shown in Table 2. Bodyfat is from Statlib which is available at (http://lib.stat.cmu.edu/.). It is the data set for estimating the percentage of body fat determined by underwater weighing and various body circumference measurements for 252 men. Bodyfat has 252 examples with 14 attributes. Other four data sets are all from UCI repository of machine learning databases which can be download from (http://www.ics.uci.edu/~mlearn/MLRepository.htm.). Housing is for concerning housing values in suburbs of Boston. It has 506 examples with 13 attributes. Mpg is used to estimate city-cycle fuel consumption in miles per gallon. There are 392 examples with 7 attributes in mpg. Pyrim and triazines are both qualitative structure activity relationships data sets. Pyrim has 74 examples with 27 attributes. Triazines has 186 examples with 60 attributes. All attributes of these data sets are scaled to the interval [-1 1].

Results: The experimental results are given in Table 3-6 where N_z denotes the number of expansion vectors. We adjust N_z according to the size of data sets in experiments to get rational results for comparison. For sinc1 and Pyrim, $N_z = 20$. For 3-D Mexican hat1 and Triazines, $N_z = 30$. For Bodyfat, $N_z = 4$. N_z of housing and mpg is 100. The number of the expansion vectors of RVM can not be chosen a priori, so we give its experimental results directly. In all tables, DSKR(R) and DSKR(K) denote DSKR with random selection initialization and DSKR with K-means clustering based initialization, respectively.

From these experimental results, we can see that initialization methods affect the performance of DSKR. Regression errors of DSKR with random selection initialization method always become bigger than these obtained by K-means clustering based selection initialization. The reason for this phenomenon is that random initialization may not lead to the uniform distribution of initial expansion vectors. However, since the difference between two kinds of regression errors is small, random selection initialization method can still be acceptable for DSKR.

Compared with ε-svr, DSKR can reduce the size of expansion vectors efficiently. For example, for the experiments on sinc1 and pyrim, the DSKR only need less than 30% percentage of ε-svr with much better results. Although v-svr can optimize N_z by adjusting parameter v, its regression errors are always bigger than DSKR and even bigger than ε-svr for the above data sets.

Table 1: Synthetic regression data sets

Data set	Function	Interval	Size
Sinc1	$y = \sin(x)/x+1$	$x \in [-10\ 10]$	67
3-D Mexican hat1	$y = \sin(\sqrt{x_1^2 + x_2^2}) / (\sqrt{x_1^2 + x_2^2}) + 1$	$x_{1,2} \in [-5\ 5]$	225

Table 2: Real-world regression data sets

Data set	Attribute	Size
Bodyfat	14	252
Housing	13	506
Mpg	7	392
Pyrim	27	74
Triazines	60	186

Table 3: Experimental result on sinc1

	snr = 10, C = 8, r = 0.04			snr = 15, C = 2, r = 0.08			snr = 25, C = 2, r = 0.08		
	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z
ε-svr	0.091	7.48	64	0.032	2.83	66	0.020	2.00	63
v-svr	0.087	8.63	20	0.061	5.23	20	0.026	2.61	20
RVM	0.118	11.38	5	0.068	6.67	5	0.021	2.15	7
DSKR(R)	0.087	7.45	20	0.036	2.84	20	0.018	1.67	20
DSKR(K)	0.085	7.41	20	0.031	2.81	20	0.015	1.61	20

Table 4: Experimental result on 3-D Mexican hat1

	snr = 10, C = 4, r = 0.04			snr = 15, C = 2, r = 0.04			snr = 25, C = 16, r = 0.04		
	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z
ε-svr	0.068	7.37	218	0.059	5.82	211	0.021	2.12	200
v-svr	0.085	7.80	30	0.061	6.11	30	0.029	3.10	30
RVM	0.103	10.80	7	0.050	5.25	10	0.017	1.83	15
DSKR(R)	0.062	6.91	30	0.051	4.77	30	0.016	1.52	30
DSKR(K)	0.059	6.84	30	0.047	4.73	30	0.014	1.51	30

Table 5: Experimental result on bodyfat, housing and mpg

	Bodyfat, C = 1, r = 0.01			Housing, C = 32, r = 0.32			Mpg, C = 32, r = 0.32		
	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z
ε-svr	0.004	0.38	9	1.33	6.57	460	1.47	6.18	349
v-svr	0.010	0.97	4	2.14	10.94	100	1.72	7.56	100
RVM	0.001	0.11	14	1.67	7.53	32	1.56	6.33	47
DSKR(R)	0.004	0.35	4	1.28	6.37	100	1.43	6.27	100
DSKR(K)	0.002	0.21	4	1.12	6.08	100	1.31	5.85	100

Table 6: Experimental result on pyrim and triazines

	Pyrim, C = 1, r = 0.02			Triazines, C = 1, r = 0.01		
	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z
ε-svr	0.025	7.30	64	0.077	23.90	173
v-svr	0.042	9.17	20	0.110	25.70	30
RVM	0.033	5.38	11	0.075	16.87	29
DSKR(R)	0.022	5.26	20	0.085	17.39	30
DSKR(K)	0.021	5.18	20	0.076	16.89	30

In most cases, RVM results in more sparse regression than DSKR. However, the regression error of RVM is usually bigger than DSKR. Given the optimal parameter C and r, the MAE and MRE of DSKR with K-means clustering based initialization are always less than those of RVM. On the other hand, the drawback of RVM is that N_z can not select a priori, in other word, N_z is only the training result of RVM. While in DSKR, we can optimize N_z according to the size of training set, i.e., N_z can be selected form 1 to N. For more expansion vectors can decrease the regression error, so DSKR has the advantage over RVM on this aspect. From Table 7-10, we also give the experimental results of RVM and DSKR with the same N_z, where the obtained results indicate that DSKR with K-means clustering based initialization usually outperforms RVM, except the data set Triazines. However, too small number of expansion vectors can not guarantee the regression performance. As we can see

Table 7: Experimental result on sinc1 of RVM and DSKR

	snr = 10, C = 8, r = 0.04			snr = 15, C = 2, r = 0.08			snr = 25, C = 2, r = 0.08		
	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z
RVM	0.118	11.38	5	0.068	6.67	5	0.021	2.15	7
DSKR (R)	0.125	11.86	5	0.074	7.35	5	0.026	2.77	7
DSKR (K)	0.112	11.14	5	0.062	6.61	5	0.019	1.97	7

Table 8: Experimental result on 3-D Mexican hat1 of RVM and DSKR

	snr = 10, C = 4, r = 0.04			snr = 15, C = 2, r = 0.04			snr = 25, C = 16, r = 0.04		
	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z
RVM	0.103	10.8	7	0.05	5.25	10	0.017	1.83	15
DSKR (R)	0.111	11.4	7	0.059	6.12	10	0.022	2.17	15
DSKR (K)	0.095	9.87	7	0.048	5.05	10	0.015	1.76	15

Table 9: Experimental result on housing and mpg of RVM and DSKR

	Housing, C = 32, r = 0.32			Mpg, C = 32, r = 0.32		
	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z
RVM	1.67	7.53	32	1.56	6.33	47
DSKR (R)	1.76	8.34	32	1.68	7.03	47
DSKR (K)	1.55	7.21	32	1.49	6.26	47

Table 10: Experimental result on pyrim and triazines of RVM and DSKR

	Pyrim, C = 1, r = 0.02			Triazines, C = 1, r = 0.01		
	MAE	MRE (%)	N_z	MAE	MRE (%)	N_z
RVM	0.033	5.38	11	0.075	16.87	29
DSKR (R)	0.036	5.64	11	0.083	17.36	29
DSKR (K)	0.031	5.37	11	0.075	16.87	29

from Table 7-10, the regression errors are much bigger than these in Table 3-6. For DSKR can optimize sparse kernel regression with arbitrary N_z, it can improve the regression performance with increasing N_z.

CONCLUSIONS

The number of support vectors plays an important role in designing a fast kernel machine. Reducing the number of support vectors can accelerate the prediction time of a kernel machine. In this research, we consider the regression problem and present a direct sparse kernel regression method named DSKR to optimize the number of support vectors in kernel regression. By adding a constraint to traditional support vector regression directly, DSKR can obtain a sparse kernel regression with user-defined number of support vectors. Experiments on two synthetic data sets and five real-world data sets show the effectiveness of DSKR. The performance of DSKR can be comparable with the state-of-the-art of other methods. DSKR may further be improved in future. For example, how to reduce the training time of DSKR is a open issue which is worthy to be solved in future.

ACKNOWLEDGMENTS

This research is supported by HongKong PolyU Grant, National 973 Key Project (grant No. 2006CB705700), 2007 National 863 project (Grant No. 2007AA1Z158), 2007 two grants from National Science Foundation of China (Grant No. 60773206, No. 60704047), 2005 National Defense Research Grant from Ministry of Education of China (Grant No. A1420461266), 2007 Cultivation Fund of the Key Scientific and Technical Innovation Project of Ministry of Education of China, New_century Outstanding Young Scholar Grant of Ministry of Education of China (Grant No. NCET-04-0496), National KeySoft Lab. at Nanjing University, The Key Lab. of Computer Science at Institute of Software, CAS, China, The Key Lab. of Computer Information Technologies at JiangSu Province, the 2004 key project of Ministry of Education of China and National Key Lab. of Pattern Recognition at Institute of Automation, CAS, China.

REFERENCES

Ben, H.A., D. Horn, H.T. Siegelmann and V. Vapnik, 2001. Support vector clustering. J. Mach. Learn. Res., 2: 125-137.

Downs, T., K.E. Gates and A. Masters, 2001. Exact simplification of support vector solutions. J. Mach. Learn. Res., 2: 293-297.

Drineas, P. and M.W. Mahoney, 2005. On the Nyström method for approximating a gram matrix for improved kernel-based learning. J. Mach. Learn. Res., 6: 2153-2175.

Gao, J. and D. Shi, 2005. Sparse kernel regression modeling based on L1 significant vector learning. In: Proceedings of the IEEE International Conference on Neural Networks and Brain, 13-15 October, Beijing, China. IEEE Express, pp: nil25-1930.

Gao, J., D. Shi and X. Liu, 2007. Significant vector learning to construct sparse kernel regression models. Neural Networks, 20: 791-798.

Keerthi, S.S., O. Chapelle and D. DeCoset, 2006. Building support vector machines with reduced classifier complexity. J. Mach. Learn. Res., 7: 1493-1515.

Lawrence, N., M. Seeger and R. Herbrich, 2002. Fast Sparse Gaussian Process Methods: The Informative Vector Machine. In: Proceeding of the 15th Neural Information Processing System, 9-14 December, Vancouver, Canada. MIT Press, pp: 606-616.

Lee, Y.J. and S.Y. Huang, 2007. Reduced support vector machines: A statistical theory. IEEE Trans. Neural Networks, 18: 1-13.

Lin, K.M. and C.J. Lin, 2003. A study on reduced support vector machines. IEEE Trans. Neural Networks, 14: 1449-1459.

Liu, D.C. and J. Nocedal, 1989. On the limited memory BFGS method for large scale optimization. Math. Progr., 3: 503-528.

Nair, P.B., A. Choudhury and A.J. Keane, 2002. Some greedy learning algorithms for sparse regression and classification with mercer kernels. J. Mach. Learn. Res., 3: 781-801.

Roth, V., 2001. Sparse kernel regressors. In: Proceedings of the International Conference on Artificial Neural Networks, 21-25 August, Vienna, Austia. Springer Press, pp: 339-346.

Schölkopf, B. and A. Smola, R.C. Williamson and P.L. Bartlett, 2000. New support vector algorithms. Neural Computation., 12: 1207-1245.

Steinwart, I., 2003. Sparseness of support vector machine. J. Mach. Learn. Res., 4: 1071-1105.

Thies, T. and F. Weber, 2006. Optimal reduced-set vectors for support vector machines with a quadratic kernel. Neur. Computation, 16: 1769-1777.

Tipping, M.E., 2001. Sparse Bayesian learning and the relevance vector machine. J. Mach. Learn. Res., 1: 211-244.

Vapnik, V., 1998. Statistical Learning Theory. Wiley, New York.

Vincent, P. and Y. Bengio, 2002. Kernel matching pursuit. Mach. Learn., 48: 165-187.

Wang, X.X., S. Chen, D. Lowe and C.J. Harris, 2006. Sparse support vector regression based on orthogonal forward selection for the generalized kernel model. NeurAL Computing, 70: 462-474.

Williams, C.K.I. and M. Seeger, 2001. Using the Nyström method to speed up kernel machines. In: Proceeding of 13th Neural Information Processing Systems, 10-14 December, Denver, USA. MIT Press, pp: 682-688.

Wu, M.R., B. Schölkopf and G. Bakir, 2006. A direct method for building sparse kernel learning algorithms. J. Mach. Learn. Res., 7: 603-624.