# A Data Reduction Algorithm Based on Support Vector Machines

Jair Cervantes[1], Farid Garcia[1], Xiaoou Li[1] and Wen Yu[2]

[1]Sección de Computación, Departamento de Ingeniería Eléctrica
CINVESTAV-IPN, Av. Instituto Politécnico Nacional 2508, 07360 México D.F., México
[2] Departamento de Control Automático, CINVESTAV-IPN, Av. Instituto Politécnico Nacional 2508, 07360 México D.F., México
Contact: yuw@ctrl.cinvestav.mx
*Paper received on 04/08/08, accepted on 10/09/08.*

**Abstract.** Recently Support Vector Machines (SVM) has been successfully applied to solve classification problems. However, the major disadvantage of SVM is slow training for classification problems with large data sets. Hence reducing the original data to contain only the support vectors is a useful goal for speeding up the training process. This reduction of training data should not affect the accuracy of SVM classifier. This paper presents a novel and efficient data reduction algorithm for SVM classifiers which can be used in order to reduce the data points when the input data is very large. In this algorithm, a first stage uses SVM classification on a small data set in order to gets a sketch of classes distribution and labels the support vectors as a data set with label +1 and the other points as a data set with label -1. The algorithm train the small data again with the new labels and obtains the new classification hyperplane. From new classification hyperplane we classify the original input data set and obtain the mos important data points, these data points are used as training data for a posterior SVM classification. The effectiveness of the approach proposed is supported by experimental results.

## 1 Introduction

In recent years Support Vector Machines (SVM) has been successfully applied to solve classification problems. Support Vector Machines are based on the idea finding a optimal classification hyperplane that divide the classes. When the data points are not linearly separable the SVM maps data points to a high dimensional feature space where a separating hyper-plane can be found. This mapping implicitly transforms the input space into another high dimensional feature space by the kernel trick. However, to find the classification hyperplane is computationally very expensive, the major disadvantage of SVM is its slow training for classification problems with large data sets, because the quadratic form is completely dense and the memory requirements grow with the square of the number of data points, so the training complexity of SVM is highly dependent on the size of a data set [6]. Hence reducing the original data to contain only the support vectors is a useful goal for speeding up the training process. This reduction of training data should not affect the accuracy of SVM classifier. Many researchers have tried to find possible methods to apply SVM

*© M. G. Medina Barrera, J. F. Ramirez Cruz, J. H. Sossa Azuela. (Eds.)*
*Advances in Intelligent and Information Technologies.*
*Research in Computing Science 38, 2008, pp. 201-211*

classification for large data sets. Generally, these methods can be divided into two types; Modify SVM classifier so that it could deal with large data sets within an acceptable time, and Reduce a large data set to smaller one so that a normal SVM could be applied. Chunking is the first decomposition method used, a standard projected conjugate gradient (PCG) chunking algorithm can scale somewhere between linear and cubic in the training set [4][8]. Dong et al introduced a parallel optimization step where block diagonal matrices are used to approximate the original kernel matrix so that SVM classification can be split into hundreds of subproblems [7].

On other hand, clustering is an effective tool to reduce data set size. For examples, hierarchical clustering [1][17], Minimal enclosing ball clustering [3] and parallel clustering [12]. Clustering based methods can reduce the computations burden of SVM, but they are very complex for large data set, because in some cases to obtain the optimal number of clusters can involve more computational cost than clustering itself [16]. Random selection is to select data in such way that the learning is maximized by the data. Sampling speeds up a classifier by randomly removing training points. Balacazar et al. [2] have used random sampling successfully to train SVM in many applications. Sampling has led to an accurate classification in their experiments with several data sets. However, has been showed that random sampling could over-simplify the training data set and lose the benefit of SVM, especially when the probability distribution of training and testing data were different.

This paper presents a novel data reduction algorithm for SVM classifiers, which can be used in order to reduce the data points when the input data is very large. According to the architecture of SVM, the training data that are near the boundaries are the most important data points. Since, the training time of SVM becomes huge when we work with large data sets, the training time can be reduced if the data that are far from the hyperplane are deleted. The proposed algorithm can delete unnecesary data from original training data and speed up the training time. In the algorithm, a first stage uses SVM classification on a small data set in order to gets a sketch of classes distribution, the algorithm labels the support vectors as a data set with label +1, this data points are the most important data in the data set and the other points as a data set with label -1, this data points are the data that are far from the hyperplane and constitute a unnecessary data set. The algorithm train the small data again with the new labels and obtains the new classification hyperplane. From new classification hyperplane we classify the original input data set and obtain the mos important data points, these data points are used as training data for a posterior SVM classification. Furthermore, this method can be applied to artificial vision for object classification, were the objects are appearance-based modeled. Appearance-Based modeling projects the images to a hiperdimentional space; such modeling demands expensive computational costs. Here we show an object classification example. The experimental results show that the accuracy obtained by proposed approach is very similar to the classic SVM methods, while the training time is significantly shorter than other SVM implementations. The rest of the paper is organized as following: Section II reviews SVM classification. Section III focuses on explaining the proposed data reduced algorithm. Section IV shows experimental results on artificial and real data sets. Conclusion is given in Section V.

## 2 Support Vector Machines

Considering binary classification, we assume that a training set $X$ is given as:

$$(\mathbf{x}_1,\mathbf{y}_1),(\mathbf{x}_2,\mathbf{y}_2),\ldots,(\mathbf{x}_n,\mathbf{y}_n) \tag{1}$$

*i.e.* $X=\{x_i,y_i\}_{i=1}^{n}$ where $x_i \in R^d$ and $y_i \in (+1,-1)$. Training SVM yields to find the optimal hyperplane,

$$y_i = sign\left[\omega^T \varphi(x_i)+b\right] \tag{2}$$

which is obtained by sole a quadratic programming problem as follows

$$\begin{aligned} &\min_{\omega,b} J(\omega)=\tfrac{1}{2}\omega^T\omega+c\sum_{i=1}^{n} \\ &subject: y_i\left[\omega^T\varphi(x_i)+b\right]\geq 1-\xi_i \end{aligned} \tag{3}$$

where $\xi_i$ are slack variables to tolerate mis-classifications $\xi_i>0$, $i=1,\ldots,n$, $c>0$. (3) is equivalent to the following quadratic programming problem which is a dual problem with the Lagrangian multipliers $\alpha_i>0$,

$$\begin{aligned} &\max_{\alpha_i} J(\omega) = -\tfrac{1}{2}\sum_{i,j=1}^{n}\alpha_i y_i \alpha_j y_j \mathbf{K}\langle x_i\cdot x_j\rangle+\sum_{i=1}^{n}\alpha_i \\ &subject : \sum_{i=1}^{n}\alpha_i y_i = 0, C\geq\alpha_i\geq 0, i=1,2,\ldots,n \end{aligned} \tag{4}$$

where $C>0$, $\alpha_i=[\alpha_1,\alpha_2,\ldots,\alpha_n]^T, \alpha_i\geq 0, i=1,2,\ldots,n$, are coefficients corresponding to $x_i, x_i$ with nonzero $\alpha_i$ is called Support Vector (SV). The function $\mathbf{K}$ is called the Mercel kernel, which must satisfy the Mercer condition [14]. The resulting optimal decision function is defined as

$$y_i = sign\left(\sum_{i=1}^{n}\alpha_i y_i \mathbf{K}(x_i,x_j)+b\right) \tag{5}$$

where $\mathbf{x}=[x_1,x_2,\ldots,x_l]$ is the input data, $\alpha_i$ and $y_i$ are Lagrange multipliers. A new object $x$ can be classified using (5). The vector $\mathbf{x}_i$ is shown only in the way of inner product. There is a Lagrangian multiplier $\alpha$ for each training point. When the maximum margin of the hyperplane is found, only the closed points to the hyperplane satisfy $\alpha>0$. These points are called support vectors *SV*, the other

points satisfy $\alpha = 0$, so the solution vector is sparse. Where $b$ is determined by Kuhn-Tucker conditions:

$$\begin{aligned} &\tfrac{\partial L}{\partial \omega} = 0, \omega = \sum_{i=1}^{n} \alpha_i y_i \varphi(x_i) \\ &\tfrac{\partial L}{\partial b} = 0, \sum_{i=1}^{n} \alpha_i y_i = 0 \\ &\tfrac{\partial L}{\partial \xi_i} = 0, \alpha_i - c \geq 0 \\ &\alpha_i \left\{ y_i \left[ \omega^T \varphi(x_i) + b \right] - 1 + \xi_i \right\} = 0 \end{aligned} \tag{6}$$

Sequential minimal optimization (SMO) breaks the large QP problem into a series of smallest possible QP problems [13]. The small QP problems can be solved analytically, which avoids using a time-consuming numerical QP optimization as an inner loop. The memory required by SMO is linear in the training set size, which allows SMO to handle very large training sets [13]. A requirement in (3) is $\sum_{i=1}^{n} \alpha_i y_i = 0$, it is enforced throughout the iterations and implies that the smallest number of multipliers can be optimized at each step is two. At each step SMO chooses two elements $\alpha_i$ and $\alpha_j$ to jointly optimize, it finds the optimal values for these two parameters while all others are fixed. The choice of the two points is determined by a heuristic algorithm, the optimization of the two multipliers is performed analytically. Experimentally the performance of SMO is very good, despite needing more iterations to converge. Each iteration uses few operations such that the algorithm exhibits an overall speedup. Besides convergence time, SMO has other important features, such as, it does not need to store the kernel matrix in memory, and it is fairly easy to implement [13].

## 3 Data reduction algorithm

To find the classification hyperplane is computationally very expensive, because the SVM need solve the problem of quadratic programming (QP) to find a separation hyper-plane which implicates a matrix of density $NxN$, where the $N$ is the number of points in the data set. Since QP's routines have quadratic complexity the SVM need huge quantities of computational time and memory for very large data sets, because the training complexity of SVM is highly dependent on the size of a data set. According to [17] to train SVM on a data set of size one million records would take years.

On the other hand, in the test phase SVM only determine the class of each data point from the weights determined at the training phase, and the complexity of this task is lineal and dependent on the support vectors obtained in the training phase. So, we can reduce the original input data set using the advantages of testing phase. Is clear that by the sparse property of SVM, the data which are not support vectors will not contribute the optimal hyperplane. The input data sets which are far away from the decision hyperplane should be eliminated, meanwhile the data sets which

are possibly support vectors should be used. According to above analysis, we make a modification on the training data set in order to reduce the original data set. In general, our approach consists of three steps which are shown in Figure 1: 1) A first stage of SVM classification on a small data set, 2) A new labeling of classes and to obtain the classification hyperplane of this new distribution 3) Data recovery, in this test phase SVM determine the class of each data point from the weights determined at the previous training phase (new distribution) from which obtain the reduced data set.

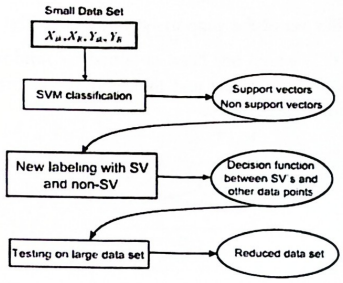



**Fig. 1.** Algorithm scheme.

### 3.1 A First Stage of SVM on a Small Data Set

In this paper, we use a special random selection algorithm to select a small percentage of data from original large data set, this algorithm has been used in [10]. We assume that $(X,Y)$ be the training patterns set, where $X=\{x_1,x_2,\ldots,x_n\}$ is the input data set, $x_i$ can be represented by a vector of $p$ dimension, i.e., $x_i=(x_{i_1},\ldots,x_{i_p})^T$, $Y=\{y_1,y_2,\ldots,y_n\}$ is the label set, label $y_i\in\{-1,1\}$. Our objective s to select a sample set $C=(c_1,c_2,\ldots,c_l)$, $l<<n$ from $X$. After random selection, we denote the data set of selected samples as $C$. We may divide $C$ into two subsets, one contains positive labels, and the other contains negative labels, i.e.,

$$C^+=\{\cup C_i \mid y=+1\}$$
$$C^-=\{\cup C_i \mid y=-1\} \tag{7}$$

Fig. 2 a) shows the original data, where the red circles represent negative labels data, and the blue squares represents positive labels data. After random selection, the selected sample data are shown in Fig. 2 b). Then, $C^-$ consists of those data which

are represented by red circles in Fig. 2 b), and $C^-$ consists of those data which are represented by blue squares in 2 b). Only these selected data will be used as training data in the first stage SVM classification. After selecting data we use a first stage of SVM classification on the small data set in order to gets a sketch of classes distribution, we use SVM classification with SMO algorithm to get the decision hyperplane

$$\sum_{k \in V_1} y_k \alpha_{1,k}^* K(x_k, x) + b_1^* \tag{8}$$

where $V_1$ is the index set of the support vectors in the first stage. Here, the training data set is $C^+ \bigcup C^-$ which has been obtained by random selection. Fig. 2 b) and c) illustrate the training data set and the support vectors obtained in the first stage SVM classification, respectively.

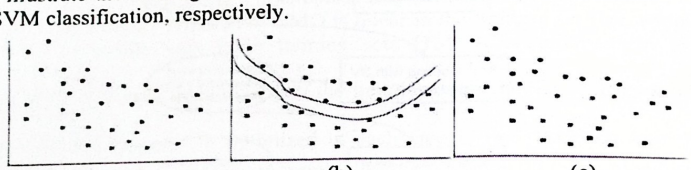



**Fig. 2.** First stage of SVM and the new labeling **(a)** Small data set **(b)** SVM classification **(c)** New labeling.

### 3.2 New Labeling

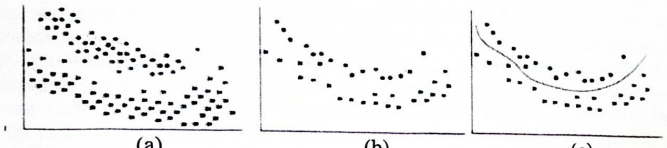
In the previous step, we have obtained a classification hyperplane, these data points are not representative enough. However, the hyperplane obtained from them gives us a reference on data distribution. From the previous step, we obtain the support vectors and non support vectors and we classify the support vectors as a data set with label +1, this data points are the most important data in the data set and the other points as a data set with label -1, this data points are the data that are far from the hyperplane and constitute a unnecessary data set. In order to obtain all the data points that are near from classification hyperplane from the original data set, the algorithm obtain a new decision hyperplane from new distribution which is given by

$$(\mathbf{x}_{a1}, \mathbf{y}_{a1}), (\mathbf{x}_{a2}, \mathbf{y}_{a2}), \ldots, (\mathbf{x}_{an}, \mathbf{y}_{an}) \tag{9}$$

*i.e.* $X = \{x_{ai}, y_{ai}\}_{i=1}^{n}$ where $x_{ai} \in R^d$ and $y_{ai} \in (+1, -1)$. where all data points with label $y_{ai} \in (+1)$ represent the support vectors obtained in the previous step and all data points with label $y_{ai} \in (-1)$ represent the non support vectors. With the purpose to find a linear separating hyperplane classifier as in Equation **(2)** we solve the QP problem, which provides the support vectors two classes. Using these SV we get a decision function which separates the positive samples (all data points

that are near from support vectors) from the negative samples (all data points that are far from support vectors). The weight vector and the threshold are $w = \sum_{i=1}^{n} \alpha_{ai} y_{ai} x_{ai}$ and $b = \frac{1}{2}\left(w^T x_{ap} + x_{an}\right)$, respectively, where $n$ is the number of support vectors, $w^T$ is the transpose of $w$, and $x_{ap}$ and $x_{an}$ are support vectors in the new distribution of the positive and the negative classes respectively. Finally the decision function of the new distribution is given by

$$\sum_{k \in V_{a2}} y_{ak} \alpha^*_{a2,k} K(x_{ak}, x_a) + b^*_{a2} \tag{10}$$



(a) (b) (c)

**Fig. 3.** Recovery of important data and final SVM classification (a) Testing on the original data set (b) Recovery of important data (c) Final SVM classification.

### 3.3 Data Recovery

Notre that, we can obtain data points that are near support vectors from the original data set. In the first stage SVM classification, the training data set is only a small percentage of the original data. The obtained decision hyperplane cannot be precise enough, in spite of this it gives us a reference on which data can be eliminated. Some useful data has not been selected during the random selection stage. So, a natural idea is to recover those data which are located between two support vectors with different label, and using the recovered data to make the classification again. In order to tune the first classification hyperplane obtained and improve the classification accuracy, the algorithm obtains all data points that are near support vectors from Equation (10). The Fig. 3 a), b) and c) show the steps which the algorithm recovers the most important data points, the recovered data set is given by

$$y_{pi} \bigcup (C_i \in V_{a2})$$

where $y_{pi}$ represent all positive data points obtained by Equation (10) and $C_i \in V_{a2}$ represent the support vectors with positive label.

## 4 Experimental results

In this Section we present the result obtained on 3 different test sets: A simple case of data set, the IJCNN data set and the Columbia object image library data set. The proposed algorithm reduce the input data set into a small data set which is composed by the most important data points. With the purpose of emphasize the utility or our approach we compared the total training time obtained with our approach with other fastr SVM implementations as simple SVM [15] and LIBSVM [5] based in SMO, all data sets are available at [5].

**Example 1** First, let consider a simple case of classification. With the purpose of clarify and emphasize the basic idea of our approach, let's consider a very simple case of classification and sectioning. We generated a set of 1,000,000 data at random, specified range and radio of aleatory generation as in [17]. Figure 4 shows the input data set with 4000 data (see a)). After the first SVM classification, a set of support vector is obtained 4 b). The Fig 4 c) shows the new labeling, the blue data points represents the support vectors obtained in the previous step, the Fig. 4 d) shows the support vectors obtained from the new labeling of classes and Fig. 4 e) the data points recovered from the original data set, finally the Fig. 4 f) shows the reduced data set.

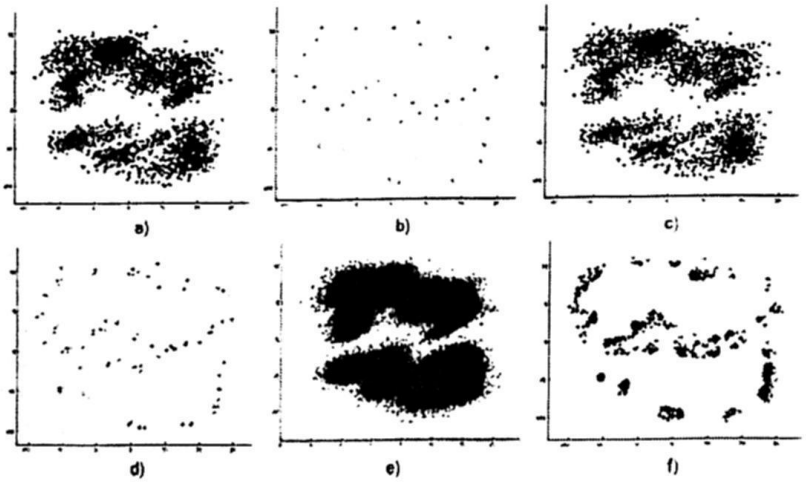



**Fig. 4.** Input data set and reduced data set.

The comparison results are shown in Table 3, where the notations are explained as follows: "#" is the data size; "t" is the training time of the whole classification which includes all the steps above mentioned; "Acc" is the accuracy; RDS is the reduced data set.

**Table 1.** Comparison of our approach with other SVM implementations.

| **GDNL Data Set** | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | Our approach | | LIBSVM | | Simple SVM | |
| # | **RDS** | **t** | **Acc** | **t** | **Acc** | **t** | **Acc** |
| 50,000 | 3532 | 59.6 | 100 | 2543 | 100 | 107.9 | 100 |
| 100,000 | 5814 | 105 | 100 | 8824 | 100 | 257.1 | 100 |
| 500,000 | 13026 | 496 | 100 | - | - | 685.6 | 100 |
| 1,000,000 | 17475 | 1029 | 100 | - | - | - | - |

In this case, the input data set has two classes as shown in Fig. 4 We use sizes of training data 1,000,000, 500,000, 100,000 and 50,000 in our experiments. We generated of separated form 50,000 testing data set, Table 1. show the results of our experiments using rbf kernel in all SVM implementations. Is clear that the proposed algorithm reduce the input data set into a small data set which is composed by the most important data points and minimize the risk of delete important data points. With the purpose of emphasize the utility of our approach we compared the total training time obtained with our approach with other SVM implementations. For 50,000 data, the proposed algorithm reduce the input data into a set with 17475 data points. The reduction of the input data points reduce considerably the training time while classification accuracy is almost the same due to algorithms nature.

**Example 2** In this example we consider a set of 3700 images that contains 19 object classes, Columbia object image library (COIL), where each image is 128×128 pixel size. Appearance-Based modeling projects all the images to a $128^2$=16384-dimention space, in other words, each image is represented as a 16384-element column vector.

**Table 2.** Comparison of our approach with other SVM implementation.

| **COIL Data Set** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Our approach | | LIBSVM | | Simple SVM | | SMO | |
| # | **RDS** | **t** | **Acc** | **t** | **Acc** | **t** | **Acc** | **t** | **Acc** |
| 3700 | 58 | 32 | 98.0 | 31 | 98.2 | 78 | 97.8 | 137 | 98.1 |

**Example 3** IJCNN 2001 The data set is available at [5]. The data set contains 49,990 training data points and 91,701 testing data points, each record has 22 attributes. The sizes of the training data we used are 1,000, 5,000, 12,500, 25,000, 37,500 and 49,990.

The comparison results are shown in table 3, where the notations are explained as follows: "#" is the data size; "t" is the training time of the whole classification which includes all the steps above mentioned; "Acc" is the accuracy; RDS is the reduced data set.

Table 3. Comparison of our approach with other SVM implementation.

| IJCNN Data Set | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Our approach | | LIBSVM | | Simple SVM | | SMO | |
| # | RDS | t | Acc | t | Acc | t | Acc | t | Acc |
| 12500 | 1975 | 105 | 97.0 | 48 | 98.2 | 183 | 97.1 | 2979 | 97.1 |
| 25000 | 3767 | 244 | 94.4 | 177 | 98.6 | 3823 | 97.2 | 6822 | 97.5 |
| 37500 | 7059 | 397 | 97.9 | 394 | 98.8 | 10872 | 97.7 | - | - |
| 49990 | 12923 | 428 | 98.5 | 731 | 98.8 | 20491 | 98.2 | - | - |

## 4 Conclusion

In this paper we present a novel and efficient data reduction algorithm for SVM classifiers which can be used in order to reduce the data points when the input data is very large. The major advantage of this algorithm is that it reduce the input data considerably, it benefits the training time for large data sets, one fast firts stage of SVM is introduced to reduce training data. A step of new labeling overcomes the drawback that only part of the original data near the support vectors are trained. Experiments done with several synthetic and real world data sets, show the proposed algorithm has great advantage when the input data set are very large.

### References


1. M. Awad, L. Khan, F.Bastani and I. L.Yen , An Effective support vector machine (SVMs) Performance Using Hierarchical Clustering, Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'04), Pages: 663-667, 2004.
2. J. L. Balcazar, Y. Dai, O. Watanabe, Provably Fast Support Vector Regression Using Random Sampling, IEEE International Conference on Data Mining, IEEE Computer Society, 2001, pp. 43-50.
3. Jair Cervantes, Xiaoou Li, Wen Yu, Support vector machine classification for large data sets via minimum enclosing ball clustering, Neurocomputing, Volume 71 , Issue 4-6, Pages 611-619, January 2008.
4. R.Collobert and S.Bengio, SVMTorch: Support vector machines for large regression problems, Journal of Machine Learning Research, Vol.1, 143-160, 2001.
5. C-C.Chang and C-J.Lin, LIBSVM: a library for support vector machines, http://www.csie.ntu.edu.tw/~cjlin/libsvm/.
6. P-H.Chen, R-E.Fan and C-J.Lin, A Study on SMO-Type Decomposition Methods for Support Vector Machines. IEEE Trans. Neural etworks, Vol.17, No.4, 893-908, 2006.
7. J-X.Dong, A.Krzyzak, and C.Y.Suen, Fast SVM Training Algorithm ith Decomposition on Very Large Data Sets. IEEE Transaction on Pattern analysis and Machine Intelligence, vol.27, no.4, pp.603-618, 2005
8. T.Joachims, Making large-scale support vector machine earning practice, Advances in Kernel Methods: Support Vector achine. MIT Press. Cambridge, MA 1998.

9. S-W.Kim and B.J.Oommen, Enhancing Prototype Reduction Schemes ith Recursion: A Method Applicable for arge ata Sets, IEEE Trans. Syst., Man, Cybern B, Vol.34, No.3, 1184-1397, 2004.
10. X. Li, J. Cervantes, and W. Yu, Two-Stage SVM Classification for Large Data Sets via Randomly Reducing and Recovering Training Data, IEEE International Conference on Systems, Man, and Cybernetics, SMC'07, Montreal, Canada, 3633-3638, 2007.
11. M.E.Mavroforakis and S.Theodoridis, A Geometric Approach to Support Vector Machine (SVM) Classification, IEEE Transactions on Neural Networks, Vol.17, No.3, 671-682, 2006.
12. C.Pizzuti and D.Talia, P-Auto Class: Scalable Parallel Clustering for Mining Large Data Sets, IEEE Trans. Knowledge and Data Eng., vol.15, no.3, pp.629-641, 2003.
13. Platt J., Fast Training of support vector machine using sequential minimal optimization. In A.S.B. Scholkopf, C. Burges, editor, Advances in Kernel Methods: support vector machine. MIT Press, Cambridge, MA 1998.
14. Vapnik V., The Nature of Statistical Learning Theory, Springer, N.Y., 1995.
15. Vishwanatan, S., Smola A., Murty M., Simple SVM, Proceedings of the Twentienth International Conference on Machine Learning. Pp 760-767. Washington, D.C., USA. 2003.
16. R.Xu and D.WunschII, Survey of Clustering Algorithms, IEEE Trans. Neural Networks, Vol.16, No.3, 645-678, 2005.
17. , H.Yu , J.Yang and J.Han , Classifying Large Data Sets Using SVMs with Hierarchical Clusters, Proc. of the 9th ACM SIGKDD 2003, Washington, DC, USA, 2003.