Neurocomputing 101 (2013) 116–128

Contents lists available at SciVerse ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Granular support vector machine based on mixed measure

Wang Wenjian [a,b,*], Guo Husheng [b], Jia Yuanfeng [b], Bi Jingye [b]

[a] Key Laboratory of Computational Intelligence and Chinese Information Processing, Shanxi University, Ministry of Education, Taiyuan 030006, PR China
[b] School of Computer and Information Technology, Shanxi University, Taiyuan 030006, PR China

ARTICLE INFO

Article history:
Received 6 May 2012
Received in revised form
10 July 2012
Accepted 5 August 2012
Communicated by G.-B. Huang
Available online 25 September 2012

Keywords:
Granular support vector machine
Model error
M_GSVM model
Mixed granule

ABSTRACT

This paper presents a granular support vector machine learning model based on mixed measure, namely M_GSVM, to solve the model error problem produced by mapping, simplifying, granulating or substituting of data for traditional granular support vector machines (GSVM). For M_GSVM, the original data will be mapped into the high-dimensional space by mercer kernel. Then, the data are divided into some granules, and those mixed granules including more information are extracted and trained by support vector machine (SVM). Finally, the decision hyperplane will be corrected through geometric analyzing to reduced model error effectively. The experiment results on UCI benchmark datasets and Interacting Proteins database demonstrate that the proposed M_GSVM model can improve the generalization performance greatly with high learning efficiency synchronously.

1. Introduction

Support vector machine introduced by Vapnik [1] is an effective method to solve pattern recognition and regression problems such as handwritten recognition, face image recognition, time series prediction, et al. At present, SVM has become a research hotspot of machine learning. In the applications of SVM, researchers pay much attention on its learning efficiency and generalization performance, and some scholars have already proposed novel approaches to improve the learning efficiency of SVM [2–8]. Although some achievements have been made, the data size in real world applications is often large and the generalization performance is largely depended on kernel function. Therefore, the researches on how to improve learning efficiency and generalization performance of SVM combining with other artificial intelligence methods still have important theoretical and practical value.

Granular computing is a new concept and computing paradigm in the domain of information processing [9]. It covers all the research about theories, methods, techniques and tools of granulation, and it can be used to process large scale information. The essence of granular computing is to find an approximate solution, which is simple and low-cost, to replace the exact solution through using inaccurate and large scale information to achieve the tractability, robustness, low cost and better describing the real world of intelligent systems or intelligent control. In a word, the combination of granular computing with intelligence computing approaches is becoming a hotspot to constitute efficient algorithms for complex problems.

To improve the performance of traditional SVM, granular support vector machine, which combines statistical learning theory and granular computing theory [10]. In general, a GSVM first creates a sequence of information granules in the original data space, and then learns on some of these granules when necessary. Finally, it aggregates information in these granules at suitable abstract level. This method cannot only obtain a better generalization for a linear separable classification problem, but also increase "linear separability" for a linear non-separable problem (or even transfer a linear non-separable problem to a totally linear separable one). Comparing with traditional SVM, the training speed of GSVM can be greatly improved and a satisfactory generalization performance can be obtained as well.

In fact, long before Tang, some other scholars have already proposed a few effective SVM models, which can be regarded as the prototype of GSVM, such as the classical "Chunking Algorithm" [1], "Decomposed Algorithm" [11], "SMO Algorithm" [12], and "LIBSVM Algorithm Library" [13].

Additionally, some scholars have already designed a number of specific GSVM algorithms, such as GSVM models based on clustering. A clustering based GSVM approach was proposed by Zhang [14]. It divides original data into a number of granules by combining commonly used clustering methods with certain evaluation rules, and it takes into classification or regression after choosing granules with more information (such as granules including more support vectors). Yu et al. [15] proposed a GSVM learning model based on hierarchical tree structure. According to the granulation results on positive and negative data, two "support vector sub trees" are constructed, respectively and those granules closing to

* Corresponding author at: Key Laboratory of Computational Intelligence and Chinese Information Processing, Shanxi University, Ministry of Education, Taiyuan 030006, PR China.
E-mail address: wjwang@sxu.edu.cn (W. Wenjian).

0925-2312/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2012.08.006

the edge are continue to extend until the desired accuracy is reached. In this way, higher learning efficiency can be obtained for large scale datasets and experiment results are witnessed in some practical applications, but the generalization performance of these models is largely determined by clustering methods.

Some GSVM models based on geometric technique are designed, such as a method based on the distance between samples and the approximate best hyperplane is proposed by Cheng et al. [16]. It has considered two geometric aspects simultaneously, the first is the distance between samples and the best approximate hyperplane and the second is the distance between the best approximate hyperplane and the obtained hyperplane. Considering the difference of granulation on kernel space and original space, an GSVM model based on kernel space is proposed by Guo et al. [17], and the rules of granulation on kernel space was given through geometric analysis. However, these approaches may be not effective for some datasets, where the distance between data cannot be measured by European distance.

Besides, Tang et al. [18] presented a GSVM model based on particle swarm optimization and it is an intuitive and easy-to-implement algorithm from the swarm intelligence community. This approach is applicable to fault classification and outperforms some previous methods. Pai et al. [19] presented a GSVM model based on fraud warning, which integrates sequential forward selection, SVM and a classification and regression tree, and it can be used to overcome information overload problems. Deb et al. [20] combine artificial neural networks with SVM. By changing the parameters of neural networks, the model can effectively reduce the dataset size and keep compressed data agree to the original data in distribution, but the interpretability of this model is absented. Tang et al. [21] presented a GSVM model based on association rules. Besides, other models such as granular support vector machine based on Rough Sets and Decision Trees are also discussed by many scholars.

All these GSVM models are granulated on the original space and trained on the kernel space (Here, these models are regarded as traditional GSVM). Although they can improve the learning efficiency, they have some losses of generalization performance. Specifically, there are mainly two reasons: first, after granulation, the data distribution may be different between those in original space and those in kernel space. Second, traditional GSVM often take granulation before training and take some informational samples (such as center of granules) to replace the whole granule when training. Therefore, data distribution errors are inevitable. These two aspects may reduce the generalization ability of GSVM [17].

This paper presents a granular support vector machine model based on mixed measure, which firstly maps the original data into a high-dimensional space to reveal the features implicit in original sample space. Then M_GSVM divides granules by some strategies like clustering, neural network, decision tree or rough set, et al., and extracts more informational mixed granules (including samples belonging to two classes) to training. Finally, the hyperplane is further corrected by geometric analyzing. Compared with traditional GSVM models, the proposed M_GSVM can largely improve the generalization performance with the high learning efficiency simultaneously.

2. Generalization performance analysis of GSVM

To better explain the M_GSVM model, we give the generalization performance analysis of traditional GSVM model firstly. Suppose the given samples set is $X=\{(x_i,y_i)\}_{i=1}^{l}$ with the independent and identically distribution $P(x,y)$, and $x_i\in R^n$, $y_i\in\{0,1\}$. k granules is produced after granulation, and new training set $X'\subseteq X$ by some samples belonging to k granules is constructed (Suppose there are l' samples in X',$l'\ll l$). The empirical risk of produced learning machine f' is

$$R_{emp}[f']=\frac{1}{l'}\sum_{i=1}^{l'}c(x_i,y_i,f'(x_i)) \tag{1}$$

here, $c(\cdot)$ is loss function. Similar with the traditional SVM model [1,22,23] and to facilitate analysis, the concept of model error is introduced firstly.

Definition 1. (Model error) In machine learning procedure, the new training set X' was produced after mapping, reconstruction, division, replacement of training set X. The optimal learning machine produced by X and X' are denoted as f and f', respectively. Model error is defined as follows.

$$E_M=\lim_{l,l'\to\infty}\left|R[f']-R[f]\right| \tag{2}$$

here, $R[\cdot]$ is expected risk of a learning machine.

Clearly, the model error can measure the classification performance difference between f and f'. For the original dataset X, after granulation, replacement and other operations, the actual training set X' may no longer follow the distribution $P(x,y)$ but a new distribution $P'(x,y)$. (In X', some data belong to X, and other data may be virtual or artificial ones. Generally, $P'(x,y)$ is different from $P(x,y)$). The actual training dataset X' and original dataset X may not meet the conditions of independent and identically distributed, and thus some training and testing data would not be classified correctly (See Fig. 1).

Then, the principle of consistency for GSVM model is introduced.

Theorem 1. (Principle of consistency for GSVM) For an GSVM model, new training dataset X' is obtained after mapping, reconstruction, granulation, replacement and other operations on training dataset X, and the corresponding optimal classifier is f'. If $|X'|\to\infty$, the sum minimization rule of empirical risk $R_{emp}[f']$ and model errors E_M is consistent to the instruction functions set F and probability distribution $P(x,y)$. That is to say,

$$\lim_{l'\to\infty}P\{\left|R[f']-(R_{emp}[f']+E_M)\right|>\varepsilon\}=0 \tag{3}$$

Proof. For an GSVM model, the distribution of X' is different with that of X, the classifier can classified almost all the actual training data correctly except. So $\lim_{l\to\infty}R_{emp}[f]-\lim_{l'\to\infty}R_{emp}[f']=0$.

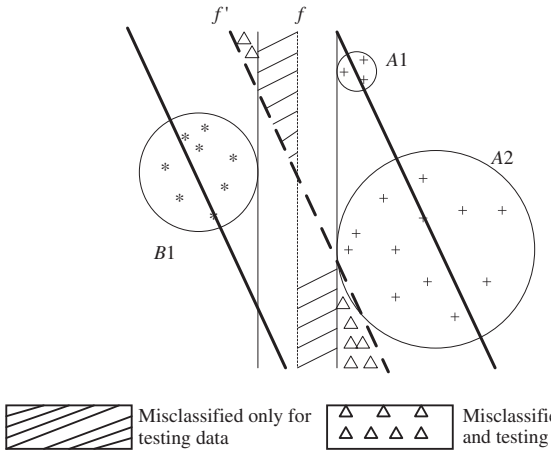


Fig. 1. Misclassified data in some regions.

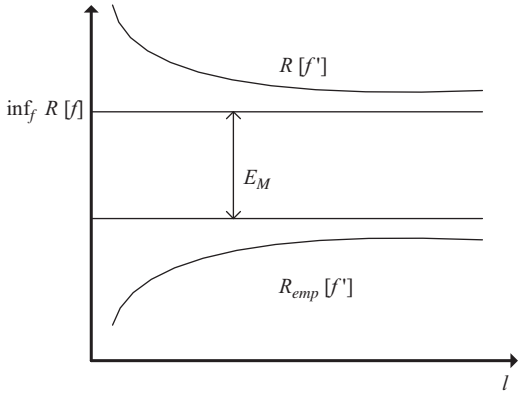


Fig. 2. Consistency principle of GSVM.

Because the distribution of the test samples of GSVM model is submitted to that of X, but not X', then,

$$\lim_{l,l'\to\infty}\left|R[f']-R_{emp}[f']\right| = \lim_{l,l'\to\infty}\left|R[f']-R[f]+R[f]-R_{emp}[f']\right|$$

$$= \lim_{l,l'\to\infty}\left|R[f']-R[f]+R_{emp}[f]-R_{emp}[f']\right| = \lim_{l,l'\to\infty}\left|R[f']-R[f]\right| = E_M$$

therefore, we can obtain the Principle of Consistency $\lim_{l'\to\infty} P\{\left|R[f']-(R_{emp}[f']+E_M)\right|>\varepsilon\}=0$ for the GSVM model.

End of proof

The presented M_GSVM is a specific GSVM model, and it is according with the principle of consistency.

Therefore, for an GSVM model, the optimal classification hyperplane f' is only suitable for X' but not X. Therefore, the generalization performance of GSVM may be greatly reduced (See Fig. 2).

To reduce the model error of traditional GSVM model, this paper will focus on improving the generalization performance of GSVM from three aspects. (1) Make the granulation and SVM training in same space to eliminate inconsistence. (2) Extract mixed granules to keep selected training samples containing support vectors as more as possible. So doing, the distribution of support vectors, deciding the new classifier f', may be close to that obtained from X after training. (3) Correct obtained hyperplane and make the final classifier as much as possible to consistent with the original dataset distribution. In this way, the model error by data operations may be reduced and the generalization performance will be improved greatly while keeping a high learning efficiency.

3. M_GSVM model

3.1. Granulation based on kernel

For a given original training set $X=\{(x_i,y_i)\}_{i=1}^{l}$, $x_i\in R^n$, and $y_i\in\{-1,1\}$ are classification labels. After nonlinear mapping Φ, the samples in high dimensional space R^N are denoted as $X=\{\Phi(x_i),y_i\}_{i=1}^{l}$. Samples are divided into k granules, that is and $X_i=\{\Phi(x_j)\}_{j=1}^{l_i}$ (l_i is the number of data in the ith granule). Each granule can be regarded as a super ball, and the center and radius are defined as follow.

Definition 2. (Center and radius of a granule super ball) Each N dimensional granule is called a granule super ball after granulation (For simplicity, the ith granule super ball corresponding to the ith granular is still denoted as X_i). The center μ_i and radius r_i of the ith granule super ball are defined as following, respectively.

$$\mu_i=\frac{1}{l_i}\sum_{p=1}^{l_i}\Phi(x_p)=\sqrt{\frac{1}{l_i^2}\left(\sum_{p=1}^{l_i}\Phi(x_p)\right)^2}$$

$$=\frac{1}{l_i}\sqrt{\sum_{p=1}^{l_i}\sum_{q=1}^{l_i}\Phi(x_p)\times\Phi(x_q)}=\frac{1}{l_i}\sqrt{\sum_{p=1}^{l_i}\sum_{q=1}^{l_i}K(x_p,x_q)} \quad (4)$$

$$r_i=\max_{x_s\in X_i}(\|\Phi(x_s)-\mu_i\|)=\max_{x_s\in X_i}\left(\sqrt{(\Phi(x_s))^2-2\Phi(x_s)\cdot\mu_i+\mu_i^2}\right)$$

$$=\max_{x_s\in X_i}\left(\sqrt{K(x_s,x_s)-\frac{2}{l_i}\sum_{p=1}^{l_i}K(x_s,x_p)+\frac{1}{l_i^2}\sum_{p=1}^{l_i}\sum_{q=1}^{l_i}K(x_p,x_q)}\right) \quad (5)$$

According to Definition 2, the distance from any $\Phi(x_j)$ to the ith granule super ball X_i in N dimensional space is

$$d(\Phi(x_j),X_i)=\sqrt{K(x_j,x_j)-\frac{2}{l_i}\sum_{p=1}^{l_i}K(x_j\cdot x_p)+\frac{1}{l_i^2}\sum_{p=1}^{l_i}\sum_{q=1}^{l_i}K(x_p,x_q)} \quad (6)$$

Because this paper focuses on designing the GSVM model with high efficient and good generalization performance in the given kernel space, how to select suitable kernel function and parameters will be not discussed, and they are described in Refs. [24–29]. In fact, the proposed M_GSVM can be combined with the existed kernel selection approaches.

Here, the granulation is accomplished iteratively by granule super balls and related measurements. The main steps of granule dividing algorithm are summarized as Algorithm 1.

3.2. Extracting mixed granules

For granules$X_1,X_2,\ldots,X_k$, we need to find mixed granules and count positive samples and negative samples in each granule. Let $positive_i=l_i^+/l_i$, $negative_i=l_i^-/l_i$, l_i^- is the number of negative samples of a granule and l_i^+ is that of positive ones. We define two parameters, *support* and *purity*, to measure the performance of granules. Let

$$support_i=l_i/l \quad (7)$$

$$purity_i=1-max(positive_i,negative_i) \quad (8)$$

when $support_i$ is greater than a given threshold (such as 0.01), granule X_i is regarded as a valid granule, otherwise, X_i is an invalid granule. if $purity_i$ is greater than a given value (such as 0.05) for a valid granule X_i, X_i is a mixed granule, otherwise, X_i is a purity granule. Then, $X_1,X_2,\ldots,X_k$ will be divided into three sets, i.e., *Set*(*invalid*), *Set*(*purity*) and *Set*(*mixed*) (See Fig. 3). In Fig. 3, $A1$ and $A2$ are purity granules, $B1$ and $B2$ are mixed granules, and $C1$ is an invalid granule. If only a sample is divided into a single granule, it will be deleted due to its *support* is too low. Obviously, if a single noise datum belonging to one class is divided into a granule, where all the rest data in this granule belong to another category, it will be judged as a purity granule and deleted due to its high purity. Therefore, introduction of *support* and *purity* can effectively avoid the impact of noise data.

As support vector information is often implicit in the *Set*(*mixed*), it can help to reduce the model error by taking all samples in the *Set*(*mixed*) and representative samples of other granules when training so as to obtain an appropriate hyperplane. If the data distribution near the hyperplane is very dense, the size of *Set*(*mixed*) will be large. We will filter mixed granules and select some large size granules to granulation repeatedly. If $l_i>2l/k$, then a mixed granule X_i is divided into $\lceil(kl_i)/(2l)\rceil$ sub granules $X_{i,1},X_{i,2},\ldots,X_{i,\lceil(kl_i)/(2l)\rceil}$

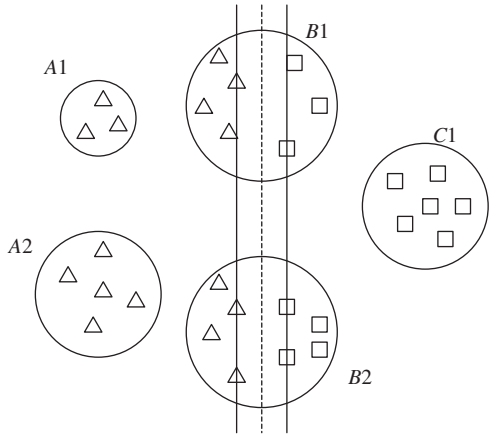


Fig. 3. Granulation schematic diagram.

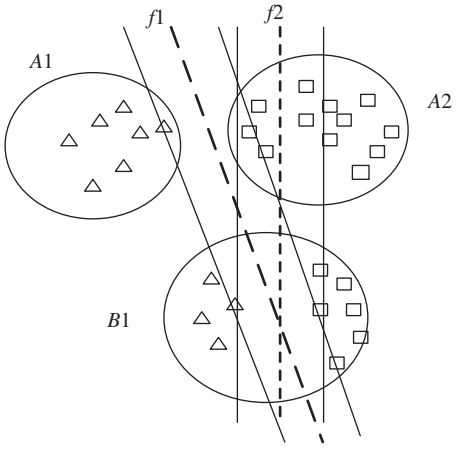


Fig. 4. Some pure granules contain support vectors.

according to the algorithm in Section 3.1 and these mixed sub granules will be added to the *Set*(*mixed*). The iteration is executed till the sizes of all mixed granules are small enough to compressing the training set effectively.

3.3. Hyperplane correction

When we obtain purity granules by using above approach, it is not equivalent to that they did not contain the original support vectors. Especially, when the distance between positive and negative samples is large, most of the positive and negative data may be divide into different granules, respectively, and this will lead to more original support vectors in purity granules (See Fig. 4). It can be seen that purity granules $A1$ and $A2$ include some support vectors, but mixed granule $B1$ only includes a little original support vectors. Generally, the purity granules containing classification information may be near to the obtained approximate hyperplane. To solve this problem, the obtained hyperplane will be corrected by applying purity granules near to obtained hyperplane with important information.

Definition 3. (Distance from a granule super ball to the hyperplane) In N dimensional space, the distance from a granule super ball X_i to the hyperplane $f: y=w\cdot\Phi(x)+b$ is defined as

$$d(X_i,f)=\frac{(w,-1)\times\mu_i^T+b}{\sqrt{w^2+1}}-r_i = \frac{\frac{1}{l_i}\sum_{k=1}^{l_i}\sum_{j\in SVs}\alpha_j y_j K(x_j,x_k)+b}{\sqrt{\sum_{\substack{j\in SVs\\ k\in SVs}}\alpha_j\times\alpha_k\times y_j\times y_k\times K(x_j,x_k)}}-r_i \quad (9)$$

here, *SVs* is the set of support vectors.

Suppose k' mixed granules in k valid granules are obtained and the number of purity granules is $(k-k')$ (The number of invalid granules is generally very small and we will ignore them here). Let $d=\min_{i\in\{\{1,\ldots,k\}/1,\ldots,k'\}}\{d(X_i,f(x))\}$, which is the distance from the nearest pure granule super ball to the hyperplane f. Those samples falling on both sides of the hyperplane within d' $(d'>d)$ margin will be taken into the training dataset (To simplify, d' takes same value in experiments, that $d'=1.25d$).

The proposed M_GSVM algorithm is summarized as Algorithm 2.

4. Model complexity analysis

4.1. Space complexity

For traditional SVM, all training data need to be put into the memory because of kernel metric computing. So space complexity of traditional SVM is $Space(SVM)=O(l^2)$. l is the number of data. Suppose the granule number of traditional GSVM is k. If granule center is used as original training data in each granule, the size of training set will be k and space complexity of it is $Space(GSVM)=O(k^2)$. Suppose the granule number of M_GSVM is also k, m is the total samples number of all mixed granules in the first training of M_GSVM, and p is number of new adding data in the retraining SVM, the samples number in final retaining SVM step is not more than $m+p$. Then, the space complexity of retraining SVM step is $O((m+p)^2)$. Because the last two processes can be implemented in sequence, the total space complexity becomes

$$Space\_complexity(M\_GSVM)\approx O((m+p)^2)+O(m^2)=O((m+p)^2) \quad (10)$$

Known that $k\le m+p<<l$, comparing with the traditional SVM model, the space complexity of M_GSVM is acceptable and closes to that of GSVM models.

4.2. Time complexity

As we know, the time complexity of SVM is $Time(SVM)=O(l^3)$, l is the number of training data. For traditional GSVM, the time complexity is $Time(GSVM)=O(k^3)$. For M_GSVM model, the time complexity of granulation step is $O(kl)$, and the time complexities of first SVM training and final retraining SVM are $O(m^3)$ and $O((m+p)^3)$, respectively. Known that usually $kl<(m+p)^3$. Therefore, the time complexity of M_GSVM model is

$$Time(M\_GSVM)\approx O(kl)+O(m^3)+O((m+p)^3)=O((m+p)^3) \quad (11)$$

The time consumption of M_GSVM is close to traditional GSVM algorithms, and both of them are better than SVM obviously.

5. Simulation experiments and discussions

The comparisons of M_GSVM model with the traditional GSVM model on generalization performance and learning efficiency are accomplished by simulation experiments, and the influence of model parameters on generalization performance is also studied.

5.1. Benchmark datasets

Ten standard datasets from UCI database (See Table 1) are used in the experiments [30]. Each dataset is averagely divided into five parts, and one of them is training set and the rests are testing sets. Cross validation is used to reduce error of experiments.

Table 1
Datasets used in experiments.

Datasets	Size	Features	Classes
Banana	8800	2	2
Titanic	13255	3	2
Thyroid	5000	5	2
Diabetic	7680	8	2
Breast_cancer	2000	9	2
Flare_solar	53300	9	2
Heart	6750	13	2
Image	11550	18	2
German	5000	20	2
Splice	15875	60	2

Table 2
Granular parameters.

Granular parameters	ART_GSVM(P)	C_GSVM, SOM_GSVM, M_GSVM(k)
Number of samples $l \le 1000$	[0.3/0.4/0.5/0.6/0.7]	[10/20/30/40/50]
Number of samples $l > 1000$	[0.5/0.6/0.7/0.8/0.9]	[20/40/60/80/100]

5.1.1. Comparisons of generalization performance

We compared M_GSVM model with traditional GSVM model based on clustering granulation (denoted as C_GSVM) and other GSVM models based on neural networks granulation (include ART_GSVM and SOM_GSVM methods). Granulation parameters of various models are shown in Table 2. *Note*, the parameter P of ART neural network is generally used to determine whether the nodes of ART neural network need to update, and it is an important factor for the performance of ART network. Because the ART_GSVM cannot set the granule number parameter directly, the parameter P of ART network will be used as the granular parameter and its value is different with other models.

Two commonly used kernel functions, Gaussian and polynomial, are used with kernel parameters 1.0 and 3, respectively, and the penalty factor C takes 1000. The testing results are shown in Tables 3 and 4. The underlined values denote the relative optimal results within the effective training time, and they can be regarded as the relative optimal results under high efficiency compared with traditional SVM model. Because the running time of traditional GSVM are not long for all granulation parameters, all maximum precision values can be regarded as relative optimal results. The running time of M_GSVM may be long in some individual cases, and they are invalid due to they cannot improve the learning efficiency. So the maximum values of M_GSVM may not be the relative optimal results, but the greater precision with high learning efficiency will take as the relative optimal results. For SVM model, only the results on *Thyroid* and *Breast_cancer* datasets are considered, and results on rest datasets are invalid because the training time is too long comparing with the mentioned four models.

It can be seen from Table 3 that when the Gaussian kernel function was used in experiments, for ART_GSVM model, three datasets, *Thyroid*, *Graman* and *Splice*, can obtain good results. For SOM_GSVM, five datasets, *Banana*, *Thyroid*, *Diabetic*, *Flare_solar* and *German* have the relative optimal results. For C_GSVM, also three datasets, *Thyroid*, *Diabetic* and *Image*, can reach the optimal results. However, except for *Image* and *Splice*, other eight datasets are still suitable for M_GSVM model.

It also can be observed from Table 4 that when the polynomial kernel function was used in experiments, for ART_GSVM model, four datasets, *Thyroid*, *Diabetic*, *German* and *splice*, can obtain good results. For SOM_GSVM, four datasets, *Diabetic*, *Thyroid*, *Titanic* and *German* have the relative optimal results. For C_GSVM, five datasets, *Banana*, *Diabetic*, *Image*, *Flare_solar* and *Thyroid*, can reach the optimal results. Similar to the experiments by Gaussian kernel, except for *Image* and *Splice*, other eight datasets are still suitable for M_GSVM model.

Comparisons of training time for four models are shown in Fig. 5 by Gaussian kernel. In Fig. 5, vertical dot lines correspond to the running time when M_GSVM obtains relative optimal results on 8 datasets. As M_GSVM cannot get relative optimal results on *Image* and *Splice*, there are not vertical dot line in (h) and (j). The traditional SVM model is not compared because it cannot obtain training results during limit time.

Also, it can be seen from Fig. 5 that on dataset *Diabetic*, the training time of M_GSVM is the shortest of all GSVM models. The efficiency of M_GSVM is similar to other models on datasets of *Banana* and *Thyroid*. The efficiency of M_GSVM is little lower than other models on datasets *Titanic*, *Breast_cancer*, *Flare_solar*, *Heart* and *German*. Overall considering learning efficiency and generalization performance (from Tables 3 and 4 and Fig. 5), the proposed M_GSVM can obtain good classification performance on majority datasets and has similar learning efficiency to the mentioned GSVM models. Similar results can also be obtained when polynomial kernel is applied.

For further analysis of the proposed M_GSVM, the prediction accuracy loss is discussed. Let the traditional SVM testing accuracy be $p(SVM)$, and the optimal accuracy of any GSVM model A in acceptable running time is denoted as $optimal(A)$. The testing accuracy loss $p(E_M)$ caused by model error E_M can be computed approximately by

$$p(E_M) \approx p(SVM) - optimal(A) \tag{12}$$

The comparison of testing accuracy losses for four GSVM models by Gaussian kernel is shown in Fig. 6. Comparing with ART_GSVM, SOM_GSVM and C_GSVM models, the prediction accuracy loss of M_GSVM is the smallest on six datasets, and it is the second small on two datasets. On *Diabetic* data, the prediction accuracy losses of SOM_GSVM, C_GSVM and M_GSVM are little differences. Only on *Splice* dataset, the prediction accuracy loss of M_GSVM model is large, and that of ART_GSVM is negative, that is to say, the generalization performance of ART_GSVM is better than original SVM model without granulation. One of the reasons may be the parameters setting. GSVM models are not always stable. If given parameters like kernel or parameter, penalization factor C, and others, can be adjusted, it will obtain satisfactory generalization performance.

The comparison about prediction accuracy losses of four GSVM models by polynomial kernel is shown in Fig. 7. Similar to Gaussian kernel, the testing accuracy loss of M_GSVM is small on most datasets. The $p(E_M)$ is little large only on *Splice* dataset.

5.1.2. Parameters tuning for M_GSVM model

Besides the granulation parameter k, there are two parameters: penalty parameter C and kernel parameter which will affect the generalization performance of M_GSVM model. To simplify the problem, only the optimization of Gaussian kernel parameter is taken into account in this experiment, while the granulation parameters are selected based on the best results (underlined values) in Table 3. Specifically, the parameters setting on different experiment datasets are shown in Table 5. The penalty parameter C is set 10, 100, 1000 and 10000, respectively, and the Gaussian kernel parameter r is set 0.1, 1.0, 1.5 and 10, respectively.

The mean testing results are shown in Table 6. The bold values are the maximal testing accuracy on different penalty and kernel parameters. Δ_1 is the difference between the maximal and the minimal testing results for different kernel parameters, and Δ_2 is the difference between the maximal and the minimal testing results for different penalty parameters C. It can be found that for different penalty parameter settings, the value of Δ_2 is always small ($\Delta_2 \le 3.54\%$). Therefore, the parameter C will hardly affect the performance for the proposed M_GSVM model. However, for different kernel parameters, the Δ_1 is relative large except dataset *Titanic*. Specially, the Δ_1 is

Table 3
Comparisons of testing results among different models by Gaussian kernel (%).

Datasets	Granulation parameter	ART_GSVM	SOM_GSVM	C_GSVM	M_GSVM	SVM
Banana	$P=0.5\|\|k=20$	68.40 ± 5.99	80.03 ± 4.16	84.02 ± 2.51	83.01 ± 3.99	
	$P=0.6\|\|k=40$	73.23 ± 5.61	77.05 ± 4.92	83.13 ± 5.10	$\mathbf{85.92 \pm 3.09}$	
	$P=0.7\|\|k=60$	70.06 ± 6.63	82.66 ± 3.15	82.99 ± 3.09	83.05 ± 1.55	–
	$P=0.8\|\|k=80$	71.26 ± 4.99	81.73 ± 1.62	81.73 ± 1.99	76.10 ± 1.64	
	$P=0.9\|\|k=100$	77.31 ± 6.09	$\mathbf{85.81 \pm 0.98}$	84.26 ± 2.38	71.73 ± 2.09	
Titanic	$P=0.5\|\|k=10$	73.29 ± 5.95	71.90 ± 9.08	73.02 ± 12.1	71.99 ± 7.31	
	$P=0.6\|\|k=20$	73.63 ± 6.50	70.49 ± 11.4	70.62 ± 10.8	71.42 ± 1.96	
	$P=0.7\|\|k=30$	73.63 ± 6.50	70.58 ± 11.4	71.00 ± 10.9	$\mathbf{74.88 \pm 3.35}$	–
	$P=0.8\|\|k=40$	73.63 ± 6.50	70.92 ± 10.0	71.00 ± 10.8	73.90 ± 3.34	
	$P=0.9\|\|k=50$	73.63 ± 6.50	71.30 ± 11.4	71.00 ± 10.8	73.61 ± 1.99	
Thyroid	$P=0.5\|\|k=20$	$\mathbf{93.03 \pm 3.53}$	88.89 ± 3.18	88.96 ± 5.37	90.99 ± 3.34	
	$P=0.6\|\|k=40$	92.08 ± 5.71	91.11 ± 3.42	91.57 ± 4.63	$\mathbf{93.00 \pm 2.70}$	
	$P=0.7\|\|k=60$	92.42 ± 5.34	91.35 ± 6.48	91.75 ± 4.62	86.30 ± 1.98	97.30 ± 0.81
	$P=0.8\|\|k=80$	92.32 ± 5.53	91.97 ± 5.37	92.28 ± 4.75	87.19 ± 2.06	
	$P=0.9\|\|k=100$	92.24 ± 5.69	$\mathbf{92.29 \pm 3.49}$	$\mathbf{92.30 \pm 5.94}$	80.01 ± 1.00	
Diabetic	$P=0.5\|\|k=20$	65.49 ± 7.76	64.20 ± 1.80	66.74 ± 2.01	70.01 ± 6.54	
	$P=0.6\|\|k=40$	67.62 ± 8.81	67.03 ± 1.21	70.31 ± 1.70	71.29 ± 11.3	
	$P=0.7\|\|k=60$	67.82 ± 8.89	71.10 ± 2.34	71.05 ± 1.94	71.90 ± 3.54	–
	$P=0.8\|\|k=80$	67.49 ± 8.12	$\mathbf{73.38 \pm 2.09}$	71.01 ± 2.02	72.24 ± 2.32	
	$P=0.9\|\|k=100$	67.10 ± 8.50	71.00 ± 1.00	$\mathbf{74.06 \pm 2.25}$	$\mathbf{73.99 \pm 1.95}$	
Breast_cancer	$P=0.5\|\|k=20$	79.85 ± 2.58	67.38 ± 3.65	74.78 ± 12.6	80.19 ± 2.13	
	$P=0.6\|\|k=40$	81.49 ± 2.38	71.66 ± 2.28	71.09 ± 3.11	$\mathbf{86.13 \pm 6.89}$	
	$P=0.7\|\|k=60$	82.38 ± 2.23	72.86 ± 1.29	74.61 ± 2.42	78.30 ± 5.43	95.63 ± 0.91
	$P=0.8\|\|k=80$	82.93 ± 2.10	74.33 ± 3.20	75.63 ± 2.58	78.73 ± 3.73	
	$P=0.9\|\|k=100$	83.21 ± 1.91	75.20 ± 2.74	78.31 ± 1.76	70.02 ± 5.05	
Flare_solar	$P=0.5\|\|k=10$	50.03 ± 5.59	55.54 ± 4.32	60.02 ± 3.71	56.17 ± 0.73	
	$P=0.6\|\|k=20$	50.35 ± 7.03	$\mathbf{62.31 \pm 5.66}$	59.37 ± 3.99	60.53 ± 2.22	
	$P=0.7\|\|k=30$	52.84 ± 7.73	56.84 ± 4.97	60.58 ± 3.78	60.16 ± 1.39	–
	$P=0.8\|\|k=40$	58.93 ± 4.98	58.63 ± 4.81	60.99 ± 4.25	61.89 ± 5.31	
	$P=0.9\|\|k=50$	61.72 ± 5.52	61.02 ± 5.54	61.19 ± 4.74	$\mathbf{62.03 \pm 2.64}$	
Heart	$P=0.5\|\|k=20$	81.78 ± 1.25	81.01 ± 2.01	80.77 ± 0.95	88.53 ± 1.70	
	$P=0.6\|\|k=40$	81.60 ± 1.06	83.09 ± 3.34	81.20 ± 1.33	87.46 ± 3.01	
	$P=0.7\|\|k=60$	81.63 ± 1.13	82.83 ± 2.58	81.82 ± 1.82	$\mathbf{92.19 \pm 2.15}$	–
	$P=0.8\|\|k=80$	82.09 ± 1.24	84.90 ± 2.99	84.01 ± 1.67	91.78 ± 1.92	
	$P=0.9\|\|k=100$	81.29 ± 1.09	82.77 ± 2.76	87.77 ± 1.00	80.09 ± 1.30	
Image	$P=0.5\|\|k=10$	68.50 ± 5.73	73.09 ± 4.33	75.84 ± 2.59	73.00 ± 5.06	
	$P=0.6\|\|k=20$	68.50 ± 5.73	77.06 ± 5.99	80.03 ± 2.36	74.53 ± 4.37	
	$P=0.7\|\|k=30$	68.50 ± 5.73	80.52 ± 4.07	79.06 ± 3.31	76.03 ± 4.12	–
	$P=0.8\|\|k=40$	68.50 ± 5.73	78.34 ± 3.76	80.93 ± 4.07	$77.42. \pm 3.38$	
	$P=0.9\|\|k=50$	68.50 ± 5.73	77.08 ± 6.50	$\mathbf{83.52 \pm 4.44}$	71.05 ± 3.99	
German	$P=0.5\|\|k=20$	71.80 ± 1.49	64.82 ± 2.01	69.06 ± 2.15	70.46 ± 1.98	
	$P=0.6\|\|k=40$	72.83 ± 1.32	68.52 ± 2.28	68.43 ± 3.00	$\mathbf{73.08 \pm 2.36}$	
	$P=0.7\|\|k=60$	$\mathbf{72.97 \pm 1.05}$	68.06 ± 1.05	71.28 ± 1.86	73.00 ± 1.99	–
	$P=0.8\|\|k=80$	72.97 ± 1.71	$\mathbf{72.47 \pm 2.21}$	71.13 ± 1.35	71.28 ± 0.98	
	$P=0.9\|\|k=100$	72.97 ± 1.53	71.38 ± 1.59	70.59 ± 1.88	71.28 ± 0.98	
Splice	$P=0.5\|\|k=10$	49.50 ± 2.51	61.25 ± 3.93	61.73 ± 4.99	50.34 ± 5.09	
	$P=0.6\|\|k=20$	49.50 ± 2.53	61.00 ± 4.09	58.30 ± 5.02	53.43 ± 5.37	
	$P=0.7\|\|k=30$	50.53 ± 4.25	60.52 ± 3.90	60.09 ± 3.49	58.32 ± 2.49	–
	$P=0.8\|\|k=40$	62.09 ± 3.30	62.24 ± 1.77	61.73 ± 2.28	56.09 ± 3.83	
	$P=0.9\|\|k=50$	$\mathbf{65.90 \pm 2.91}$	60.03 ± 3.33	60.00 ± 3.13	57.66 ± 3.99	

larger than 10% on datasets *German* and *Banana*, and is larger than 60% on dataset *Splice*. For the dataset *Splice*, the testing accuracy is only about 5% when the kernel parameter is 0.1. The reason may be the "poor" kernel feature space where the data cannot be classified well. So the kernel parameter may produce more effect on testing results comparing with the penalty parameter on most datasets. In practical applications, we can select suitable kernel function and parameter by some other methods. In following experiments, the Gaussian kernel parameter is selected as 1.0 for simplicity.

5.1.3. Effectiveness verify of M_GSVM model

5.1.3.1. Effectiveness evaluation factors. Suppose support vectors set obtained by traditional SVM on training set X is $SVs=\{sv_1,sv_2,\ldots,sv_t\}$. The training dataset of GSVM is X' with l' samples after mapping, simplify, granulation, replacement and other operations. Let $co=SVs\cap X'$. The samples compress rate (*compress_rate*) and support vector overcast rate (*overcast_sv*) are defined, respectively.

$$compress\_rate = 1-l'/l \tag{13}$$

$$overcast\_sv = |co|/|SVs| = |co|/t \tag{14}$$

here, $|\bullet|$ represents the number of samples belonging to the set. In general, from the view point of generalization performance and learning efficiency, if the *compress_rate* of an GSVM model is high, its learning efficiency is also high. And the bigger *overcast_sv*, the better generalization performance. The main factor affecting them is granulation parameter. The *compress_rate* of ART_GSVM, SOM_GSVM and C_GSVM can be calculated easily according to formula (13).

Table 4
Comparisons of testing results among different models by polynomial kernel (%).

Data sets	Granulation parameter	ART_GSVM	SOM_GSVM	C_GSVM	M_GSVM	SVM
Banana	$P=0.5 \| \| k=20$	62.25 ± 2.38	78.12 ± 2.53	83.17 ± 3.02	84.03 ± 4.52	
	$P=0.6 \| \| k=40$	72.50 ± 3.56	75.73 ± 4.08	83.53 ± 4.28	$\mathbf{86.91 \pm 3.70}$	
	$P=0.7 \| \| k=60$	70.35 ± 5.93	77.10 ± 3.25	80.93 ± 3.92	82.59 ± 2.54	–
	$P=0.8 \| \| k=80$	72.19 ± 5.31	80.59 ± 2.21	82.52 ± 3.45	80.48 ± 3.66	
	$P=0.9 \| \| k=100$	72.71 ± 3.90	82.34 ± 2.24	$\mathbf{85.53 \pm 3.66}$	73.54 ± 3.01	
Titanic	$P=0.5 \| \| k=20$	73.29 ± 5.95	72.65 ± 5.81	72.71 ± 9.58	72.53 ± 3.02	
	$P=0.6 \| \| k=40$	73.29 ± 5.95	73.42 ± 2.27	68.32 ± 10.0	71.87 ± 2.55	
	$P=0.7 \| \| k=60$	73.63 ± 6.50	73.35 ± 3.94	69.94 ± 8.70	73.94 ± 2.08	–
	$P=0.8 \| \| k=80$	73.63 ± 6.50	$\mathbf{74.45 \pm 3.55}$	71.00 ± 10.8	$\mathbf{74.10 \pm 3.51}$	
	$P=0.9 \| \| k=100$	73.63 ± 6.50	73.96 ± 2.53	71.00 ± 10.8	73.00 ± 2.94	
Thyroid	$P=0.5 \| \| k=20$	$\mathbf{93.53 \pm 4.00}$	89.73 ± 2.40	90.05 ± 3.73	92.75 ± 2.24	
	$P=0.6 \| \| k=40$	92.87 ± 4.71	92.37 ± 3.19	91.76 ± 2.99	$\mathbf{93.23 \pm 2.70}$	
	$P=0.7 \| \| k=60$	91.76 ± 6.04	92.40 ± 5.56	92.48 ± 3.89	90.83 ± 1.98	98.21 ± 1.53
	$P=0.8 \| \| k=80$	91.54 ± 2.25	91.00 ± 2.18	92.10 ± 5.17	88.54 ± 2.06	
	$P=0.9 \| \| k=100$	91.54 ± 3.87	$\mathbf{92.46 \pm 3.82}$	$\mathbf{93.35 \pm 4.31}$	83.00 ± 1.00	
Diabetic	$P=0.5 \| \| k=20$	70.13 ± 5.81	67.51 ± 2.11	68.42 ± 2.73	70.01 ± 5.99	
	$P=0.6 \| \| k=40$	71.35 ± 5.54	73.00 ± 3.24	69.95 ± 2.15	70.94 ± 3.05	
	$P=0.7 \| \| k=60$	73.65 ± 6.20	72.47 ± 1.94	71.81 ± 2.54	72.01 ± 2.54	–
	$P=0.8 \| \| k=80$	69.84 ± 6.43	$\mathbf{73.73 \pm 3.36}$	72.98 ± 3.19	72.79 ± 2.88	
	$P=0.9 \| \| k=100$	$\mathbf{74.26 \pm 7.97}$	71.55 ± 2.50	$\mathbf{73.56 \pm 2.64}$	$\mathbf{74.08 \pm 3.06}$	
Breast_cancer	$P=0.5 \| \| k=20$	78.07 ± 2.99	68.52 ± 1.97	76.73 ± 2.25	83.13 ± 7.51	
	$P=0.6 \| \| k=40$	80.52 ± 2.20	66.31 ± 2.66	75.37 ± 3.84	$\mathbf{85.65 \pm 4.44}$	
	$P=0.7 \| \| k=60$	82.39 ± 2.54	72.03 ± 2.90	75.94 ± 2.61	81.50 ± 2.80	93.99 ± 3.37
	$P=0.8 \| \| k=80$	82.00 ± 1.19	69.56 ± 3.04	77.85 ± 2.06	77.47 ± 3.83	
	$P=0.9 \| \| k=100$	83.21 ± 2.77	66.83 ± 2.99	79.48 ± 4.07	74.36 ± 5.25	
Flare_solar	$P=0.5 \| \| k=20$	55.21 ± 1.20	58.23 ± 3.89	58.35 ± 4.02	55.28 ± 2.95	
	$\mathrm{P}=0.6 \| \| k=40$	55.21 ± 1.20	60.48 ± 4.37	60.73 ± 3.85	58.42 ± 3.10	
	$P=0.7 \| \| k=60$	55.21 ± 1.20	60.02 ± 3.36	60.02 ± 3.97	59.53 ± 3.21	–
	$P=0.8 \| \| k=80$	60.53 ± 3.35	58.56 ± 4.03	61.83 ± 2.12	61.10 ± 2.78	
	$P=0.9 \| \| k=100$	60.02 ± 2.78	61.53 ± 2.97	$\mathbf{62.06 \pm 5.48}$	$\mathbf{62.56 \pm 5.14}$	
Heart	$P=0.5 \| \| k=20$	79.30 ± 1.29	79.97 ± 2.98	79.54 ± 2.28	89.54 ± 2.28	
	$P=0.6 \| \| k=40$	82.07 ± 0.76	81.58 ± 2.37	81.66 ± 2.73	90.73 ± 1.16	
	$P=0.7 \| \| k=60$	84.56 ± 0.94	84.59 ± 3.50	82.39 ± 2.24	$\mathbf{92.64 \pm 2.93}$	–
	$P=0.8 \| \| k=80$	84.93 ± 1.10	83.73 ± 2.86	86.11 ± 3.60	90.00 ± 3.50	
	$P=0.9 \| \| k=100$	82.75 ± 1.88	81.25 ± 3.05	88.59 ± 3.08	85.35 ± 2.78	
Image	$P=0.5 \| \| k=20$	65.35 ± 4.26	75.64 ± 3.58	79.54 ± 3.77	71.59 ± 4.85	
	$P=0.6 \| \| k=40$	68.50 ± 5.73	78.23 ± 4.02	78.29 ± 2.00	73.26 ± 3.73	
	$P=0.7 \| \| k=60$	68.50 ± 5.73	79.94 ± 3.91	80.73 ± 5.31	74.63 ± 3.35	–
	$P=0.8 \| \| k=80$	68.50 ± 5.73	76.81 ± 5.13	81.06 ± 2.54	$78.56. \pm 2.10$	
	$P=0.9 \| \| k=100$	68.50 ± 5.73	74.52 ± 4.94	$\mathbf{81.77 \pm 3.06}$	75.55 ± 3.94	
German	$P=0.5 \| \| k=20$	70.94 ± 1.21	65.05 ± 1.97	67.30 ± 2.25	71.51 ± 2.06	
	$P=0.6 \| \| k=40$	71.83 ± 1.57	67.35 ± 2.54	66.48 ± 3.17	72.94 ± 2.79	
	$P=0.7 \| \| k=60$	71.14 ± 2.16	66.62 ± 2.12	70.85 ± 2.07	$\mathbf{73.53 \pm 2.15}$	–
	$P=0.8 \| \| k=80$	72.97 ± 1.05	69.73 ± 1.68	71.19 ± 1.98	72.08 ± 1.54	
	$P=0.9 \| \| k=100$	$\mathbf{72.97 \pm 1.05}$	$\mathbf{73.14 \pm 2.97}$	70.07 ± 2.31	70.06 ± 1.30	
Splice	$P=0.5 \| \| k=20$	53.30 ± 1.94	59.73 ± 3.82	59.48 ± 3.00	50.28 ± 3.98	
	$P=0.6 \| \| k=40$	53.30 ± 1.94	60.48 ± 3.66	58.72 ± 2.94	55.45 ± 4.06	
	$P=0.7 \| \| k=60$	10.76 ± 2.53	62.54 ± 4.00	57.54 ± 2.56	57.37 ± 2.77	–
	$P=0.8 \| \| k=80$	58.92 ± 3.14	61.07 ± 3.45	61.02 ± 2.78	55.10 ± 3.54	
	$P=0.9 \| \| k=100$	$\mathbf{64.09 \pm 2.77}$	59.40 ± 2.65	61.17 ± 3.53	53.66 ± 4.49	

For ART_GSVM, l' is the determined by parameter P, while for SOM_GSVM and C_GSVM, l' is the number of granules k. For M_GSVM, l' refers to $m+p$, the sample number in final retaining SVM step. The *overcast_sv* of several models can be computed directly by Eq. (14).

In order to improve learning efficiency, the size of training data should be reduced as more as possible. On the other hand, it should also reserve enough support vectors. Hence, a new factor, support vectors relative retention rate ρ, is introduced.

$$\rho = \frac{overcast\_sv}{compress\_rate} \tag{15}$$

During acceptable training time, if ρ is big, more original support vectors are retained and the generalization performance is better. Conversely, the model error will be large.

5.1.3.2. Experiments on effectiveness of M_GSVM. For ART_GSVM, SOM_GSVM and C_GSVM models, those samples which are the nearest to the granule centers are usually used as training data, and then the sample compress rates show little be difference among these three models. Therefore, for simplicity, the M_GSVM will be only compared with C_GSVM. Because experiment results are similar by Gaussian and polynomial kernels, only the Gaussian kernel is taken into account in this experiment. When C_GSVM and M_GSVM models take maximum test accuracy (from Table 3), the corresponding *compress_rate*, *overcast_sv* and ρ are listed in Table 7. $compress\_rate_1$, $overcast\_sv_1$ and ρ_1 represent sample compress rate, support vectors overcast rate and support vectors relative retention rate after the first training of M_GSVM model, and $compress\_rate_2$,$overcast\_sv_2$ and ρ_2 represent those three factors after retraining SVM.

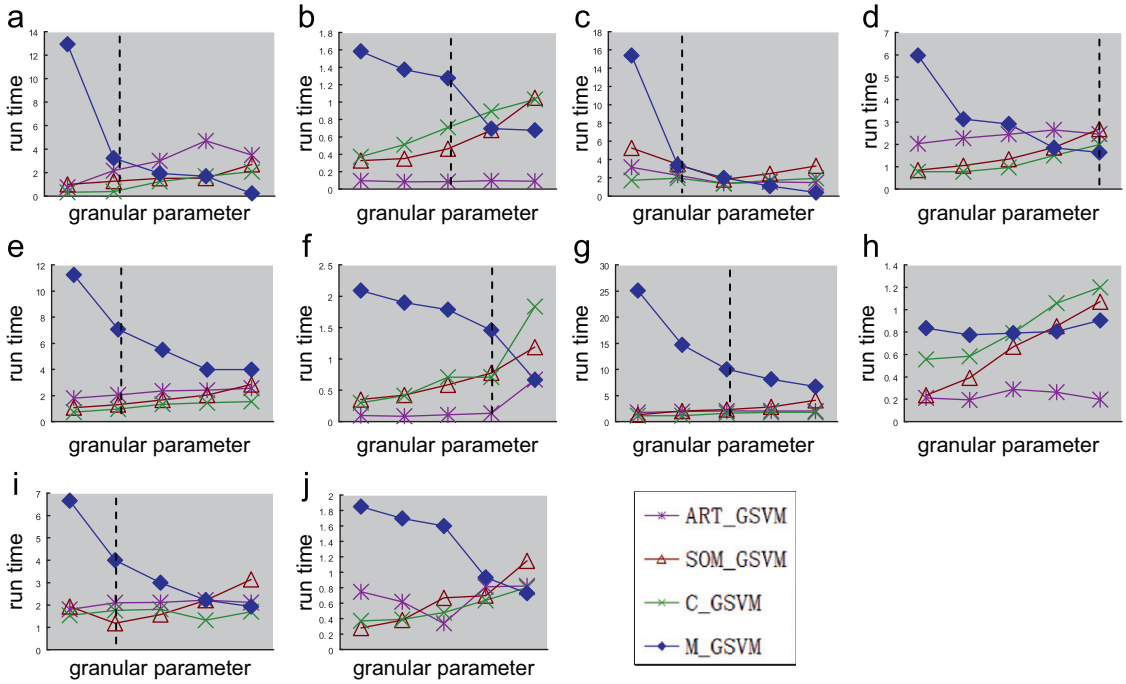


Fig. 5. Comparisons of training time for four models by Gaussian kernel. (a) Banana, (b) Titanic, (c) Thyroid, (d) Diabetic, (e) Breast_cancer, (f) Flare_solar, (g) Heart, (h) Image, (i) German and (j) Splice.

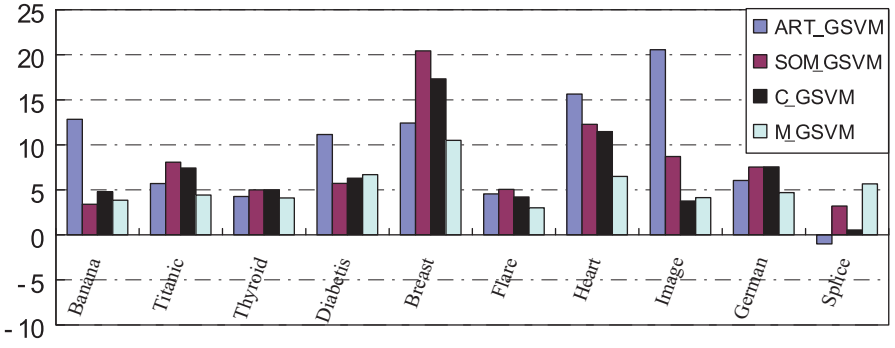


Fig. 6. Comparisons of $p(E_M)$ for four GSVM models (RBF kernel).

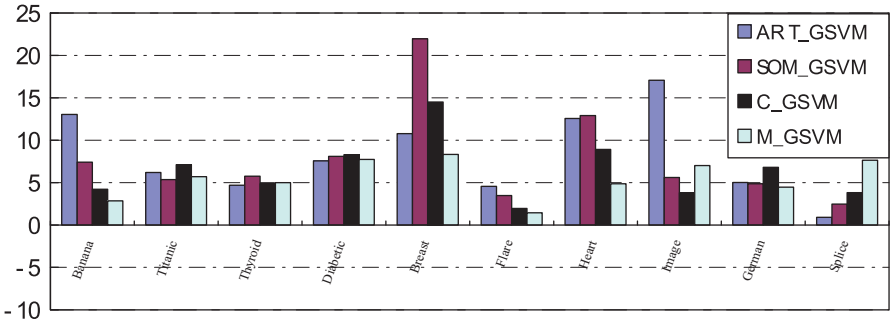


Fig. 7. Comparisons of $p(E_M)$ for four GSVM models (Polynomial kernel).

It can be seen from Table 7 that ρ_2 of M_GSVM model after hyperplane correction is greater than that of C_GSVM only on five datasets, *Banana*, *Titanic*, *Thyroid*, *Breast_cancer* and *Heart*. However, the support vectors overcast rate of M_GSVM on all datasets is higher than that of C_GSVM model obviously. The reason is that many more support vectors are lost in the process of mapping, simplifying, granulation and other operations of C_GSVM. Although the loss of some support vector information in positive and negative granules may be offset, it will be difficult to make further improvements of C_GSVM model if the information is lost at previous steps. While, the M_GSVM model has the high support vectors overcast rate due to using mixed measure and hyperplane correction.

Fig. 8 shows the relationship between granulation parameters and support vectors relative retention rate ρ_1 and ρ_2 in two stages for M_GSVM model. In Fig. 8, vertical dot lines correspond to the number of granules when the M_GSVM model reaches the biggest support vectors relative retention rate. It can be seen that the optimal number of granules in Fig. 8 is the same as that in Fig. 5 when the M_GSVM obtains the relative optimal generalization performance on 6 datasets, *Banana*, *Titanic*, *Thyroid*, *Breast_cancer*,

Table 5
Parameters setting of M_GSVM model.

Data sets	Granular parameter k	Penalty parameter C	Gaussian kernel parameter r
Banana	40	10/100/1000/10000	0.1/1.0//1.5/10
Titanic	30		
Thyroid	40		
Diabetic	100		
Breast_cancer	40		
Flare_solar	50		
Heart	60		
Image	40		
German	40		
Splice	30		

Flare_solar, *Heart*. So for most datasets, when the generalization performance is good, the corresponding ρ may be high. It also can be observed that the ρ takes a high value when the number of granules is suitable except for *Image* dataset. Generally, when k is small, the number of obtained granule super balls is small, and then many granule super balls are regarded as mixed granules. This will lead to the *overcast_sv*$_1$ and ρ_1 too small. However, when granulation parameter is large, the number of obtained granule super balls is large, and too many granule super balls as purity granules will be deleted. This will result in the *overcast_sv* and ρ_1 too low. For example, when granulation parameter k takes 40 on *Thyroid* and 60 on *Dia*betic, although ρ_2 is slightly less than ρ_1, the *overcast_sv*$_2$ is obviously larger than the *overcast_sv*$_1$. It is said that some non support vector information is added, but the number of valid

Table 6
Testing results on different parameters (%).

Datasets	Penalty parameter	$r=0.1$	$r=1.0$	$r=1.5$	$r=10$	Δ_1
Banana	10	83.96	84.37	84.69	56.77	27.92
	100	84.76	85.25	84.52	56.20	29.05
	1000	85.32	**85.92**	84.69	56.38	29.54
	10000	84.29	84.10	82.06	58.24	26.05
	Δ_2	1.36	1.82	2.63	2.04	
Titanic	10	70.59	72.53	74.13	71.15	3.54
	100	71.28	72.94	74.06	70.93	3.13
	1000	73.94	74.88	**74.94**	74.35	1.00
	10000	71.34	71.34	71.06	71.34	0.28
	Δ_2	3.35	3.54	3.88	3.42	
Thyroid	10	89.57	90.06	91.50	83.18	8.32
	100	89.69	91.58	92.37	81.69	10.68
	1000	91.23	93.00	**93.51**	81.50	12.01
	10000	88.62	88.94	87.91	83.14	5.80
	Δ_2	2.61	2.64	5.60	1.68	
Diabetic	10	70.51	71.59	72.00	74.18	3.67
	100	70.48	71.94	73.52	74.82	4.34
	1000	70.69	73.99	74.15	75.16	4.47
	10000	70.97	72.53	73.47	**75.48**	4.51
	Δ_2	0.49	2.40	2.15	1.30	
Breast_cancer	10	83.58	84.93	83.56	82.58	2.35
	100	84.06	85.19	84.10	83.16	2.03
	1000	84.40	**86.13**	84.11	83.47	2.66
	10000	82.73	83.09	81.70	82.09	1.39
	Δ_2	1.33	3.04	2.41	1.38	
Flare_solar	10	59.81	61.57	60.33	64.47	4.66
	100	59.34	62.19	61.10	65.32	5.98
	1000	59.98	62.03	61.28	**66.59**	6.61
	10000	57.60	61.48	59.94	63.20	5.60
	Δ_2	2.38	0.71	1.34	3.39	
Heart	10	84.95	91.21	89.25	90.15	6.26
	100	87.03	91.58	90.53	91.20	4.55
	1000	86.15	**92.19**	91.10	91.49	6.04
	10000	85.78	90.27	89.35	91.13	5.35
	Δ_2	2.08	1.92	1.85	1.34	
Image	10	74.32	76.91	77.01	81.13	6.81
	100	76.10	77.25	77.53	**84.28**	8.18
	1000	75.94	77.42.	77.66	82.96	7.02
	10000	75.52	76.83	76.94	83.30	7.78
	Δ_2	1.78	0.59	0.72	3.15	
German	10	58.28	72.72	73.55	74.96	16.68
	100	59.40	72.38	74.08	75.13	15.73
	1000	61.37	73.08	74.01	**76.48**	15.11
	10000	60.59	71.91	74.14	75.00	14.41
	Δ_2	3.09	1.17	0.59	1.52	
Splice	10	4.49	57.39	57.53	76.38	71.89
	100	4.49	58.03	57.84	**77.59**	73.10
	1000	4.49	58.32	58.22	75.42	70.93
	10000	5.48	56.83	56.51	75.07	69.59
	Δ_2	0.99	1.49	1.71	2.52	

Table 7
Three factors of C_GSVM and M_GSVM models.

	Datasets	*Banana*	*Titanic*	*Thyroid*	*Diabetic*	*Breast_cancer*	*Flare_solar*	*Heart*	*Image*	*German*	*Splice*
C_GSVM	*compress_rate*	0.038	0.025	0.025	0.034	0.023	0.019	0.024	0.056	0.030	0.050
	overcast_sv	0.189	0.132	0.132	0.373	0.194	0.125	0.170	0.530	0.383	0.500
	ρ	4.97	5.28	5.28	10.97	8.43	6.58	7.08	9.46	12.77	10.00
M_GSVM	$compress\_rate_1$	0.082	0.096	0.057	0.051	0.074	0.075	0.049	0.084	0.076	0.072
	$overcast\_sv_1$	0.507	0.434	0.593	0.33	0.538	0.466	0.701	0.627	0.793	0.79
	ρ_1	6.18	4.52	10.4	6.471	7.27	6.21	14.31	7.46	10.43	10.97
	$compress\_rate_2$	0.118	0.121	0.099	0.055	0.103	0.148	0.063	0.101	0.098	0.087
	$overcast\_sv_2$	0.713	0.662	0.865	0.53	0.894	0.751	0.772	0.854	0.812	0.835
	ρ_2	6.04	5.47	8.74	9.636	8.68	5.07	12.25	8.46	8.29	9.6
$overcast\_sv$:$overcast\_sv_2$		1:3.77	1:5.02	1:6.55	1:1.42	1:4.6	1:6.01	1:4.54	1:1.61	1:2.12	1:1.67
ρ:ρ_2		1:1.22	1:1.04	1:1.66	1:0.88	1:1.03	1:0.77	1:1.73	1:0.89	1:0.65	1:0.96

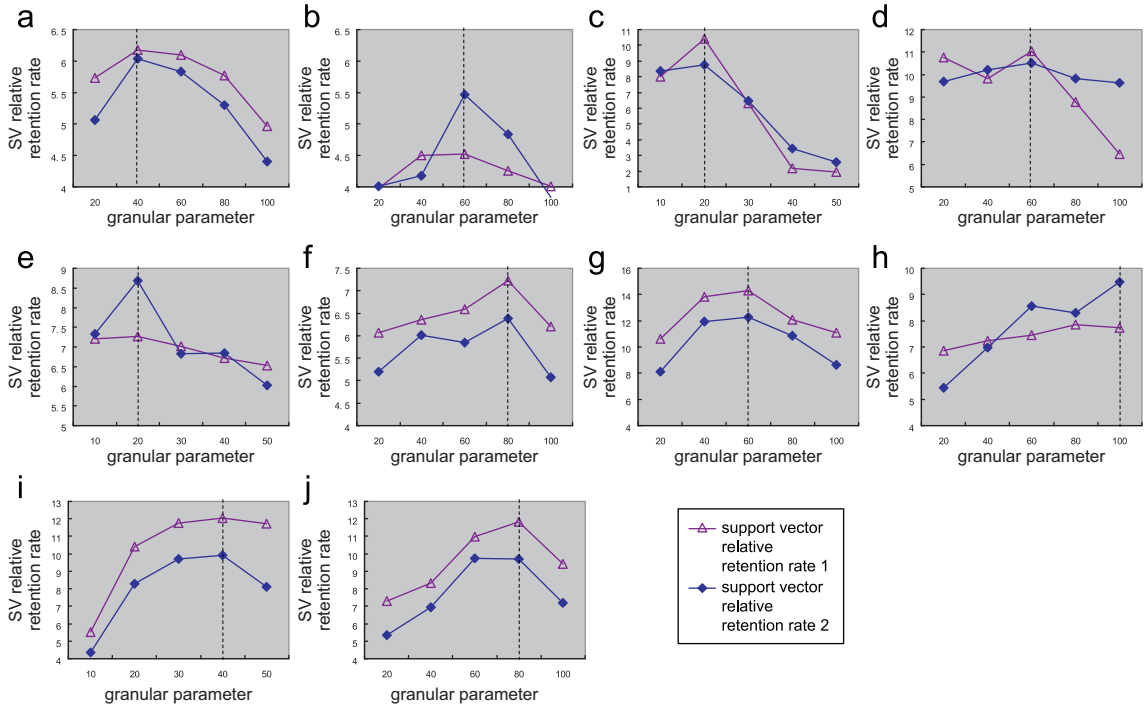


Fig. 8. Tendency of support vector relative retention rate for M_GSVM model. (a) Banana, (b) Titanic, (c) Thyroid, (d) Diabetic, (e) Breast_cancer, (f) Flare_solar, (g) Heart, (h) Image, (i)German, (j) Splice.

original support vectors is increased. Therefore, purity granules may contain useful support vector information for SVM learning and the hyperplane correction is effective. In these experiments, when granulation parameter takes those values near the vertical dot lines, final training samples will include more original support vectors and they will help to improve the model so as to obtain approximate generalization performance of traditional SVM.

5.2. Database of Interacting Proteins

The M_GSVM model is further verified on Proteins dataset. Database of Interacting Proteins (DIP) is applied to predict the relationship of interacting proteins, and it can be download from http://dip.doe-mbi.ucla.edu/dip/Main.cgi. This database consists of the interaction of proteins from various species, such as *D. melanogaster*, *S. cerevisiae*, *E.coli*, *C. elegans*, *H. sapiens*, *H. pylori*, *M. musculus and R. norvegicus*. The interactions of each species include full and core. Among all those interactions, all the core interactions are verified by biological experiments, while the full interactions have not been tested by experiments yet. The data only including core interactions of *Saccharomyces cerevisiae* (baker's yeast) updated on Oct. 10th, 2010 are used in this experiment.

The used dataset consists of 4514 pairs of proteins, each of which has its own ID. With the help of this ID, we can successfully find out the sequence of the amino acid of the corresponding protein from another database (*FULL*, a database contains the sequence of amino acid of different protein). Then, the sequence is coded according to segment local description. After this process, each protein has 630 features, and the database consists of 4514 samples, each of which has two proteins, and each sample will have 1260 features. 2000 pairs of them are positive (two protein have relations) and the others are negative (they have no relations). In our experiments, the principal component analysis (PCA) method is adopted to extract features, and 10, 20, 30, 40, 50 principal components are selected, respectively. Four models, SVM, SOM_GSVM, C_GSVM and M_GSVM, are compared, and the experiment results are shown in Table 8.

In Table 8, the bold values denote the maximum prediction accuracy and the corresponding running time under different granulation parameters for each method. Comparing with traditional SVM, the efficiency of any GSVM has been improved at least 10 times. It can be observed that the testing results of SVM are the best in all the cases, but the training times are the longest. For other three GSVM models, when number of principal components is selected as 10, 30, 40 and 50, the M_GSVM is the best. When 20

Table 8
Comparison of experiment results among different models on database of Interacting Proteins.

No. of principal components	Experiment Results	SVM	C_GSVM (k)					SOM_GSVM (k)					M_GSVM (k)				
			100	200	300	400	500	100	200	300	400	500	100	200	300	400	500
10	Accuracy (%)	**87.066**	69.579	80.791	81.360	**83.324**	82.470	72.073	77.394	79.941	80.582	**80.852**	81.029	**84.953**	83.145	81.177	79.855
	Time(s)	***1483.6***	0.4063	2.0625	5.7344	***11.875***	21.266	8.312	18.218	35.500	68.875	***104.44***	25.438	***45.039***	69.737	98.142	93.281
	#SV	785	98	197	274	353	424	94	187	271	365	456	398	426	437	513	497
20	Accuracy (%)	**88.902**	71.400	80.450	81.574	**84.633**	84.505	72.813	81.364	80.639	83.214	**84.537**	79.693	**84.628**	82.390	81.706	80.784
	Time(s)	***1446.0***	0.3906	2.0625	5.6875	***12.125***	22.375	8.500	19.719	49.938	84.471	***128.38***	13.183	***41.149***	52.377	79.828	91.078
	#SV	1124	100	200	299	392	478	100	200	297	396	494	411	449	501	527	483
30	Accuracy (%)	**84.633**	69.465	75.740	78.543	**80.322**	79.681	70.593	77.281	75.659	**79.728**	79.514	**83.560**	81.723	80.370	78.094	79.349
	Time(s)	***1454.3***	0.3906	2.1250	5.7969	***12.453***	22.578	11.750	24.228	50.719	***86.065***	123.83	***21.296***	33.628	25.730	54.096	71.923
	#SV	1268	100	200	300	398	497	100	200	300	400	499	391	477	469	532	541
40	Accuracy (%)	**81.460**	68.910	73.833	75.484	**77.448**	74.189	68.914	71.546	75.815	**76.071**	75.772	**78.312**	76.438	77.355	74.299	69.954
	Time(s)	***1483.7***	0.4375	2.1875	5.8438	***12.656***	22.656	15.656	27.750	50.312	***89.340***	124.99	***19.373***	31.078	30.095	58.433	109.32
	#SV	1335	100	200	300	399	496	100	200	300	399	500	402	467	471	543	559
50	Accuracy (%)	**78.970**	65.737	69.479	71.514	**72.994**	72.694	70.067	70.622	**74.161**	73.852	73.89	**75.970**	73.899	74.014	71.344	70.967
	Time(s)	***1454.7***	0.4219	2.1250	5.7344	***12.047***	22.188	15.594	30.656	***57.047***	92.691	143.39	***87.376***	95.682	138.49	233.35	159.82
	#SV	1391	100	200	300	400	499	100	200	300	400	500	431	479	466	559	596
1260	Accuracy (%)	**69.374**	53.558	59.632	**63.094**	60.380	57.533	–	–	–	–	–	**61.353**	58.075	57.410	55.832	52.044
	Time(s)	***3157.4***	59.851	86.560	***173.8***	306.40	389.37	–	(training time is too long)			–	***207.40***	371.34	692.58	1138.4	71.90
	#SV	2537	100	200	300	400	500	–	–	–	–	–	537	528	694	842	731

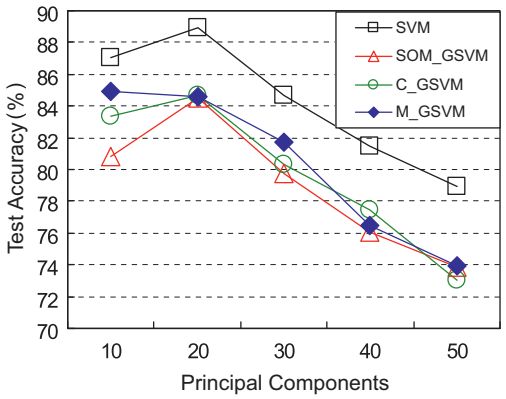


Fig. 9. Changing tendency of prediction accuracy with principle components.

and 1260 principal components are selected, the M_GSVM is only inferior to C_GSVM but with very little difference. For SOM_GSVM model, when all the 1260 features are used, the distance between any two samples may be very large and the similarity of samples are not measured effectively. Then, training and testing results cannot be obtained. Although the running time of M_GSVM is a little longer than that of C_GSVM, M_GSVM has significant improvement for generalization performance in most cases comparing with C_GSVM and SOM_GSVM. This means that M_GSVM can retain most of the original support vectors and reduce the model error. Hence, M_GSVM can obtain almost the same generalization performance like standard SVM and make good trade-off between learning efficiency and generalization performance.

Fig. 9 is the changing tendency of the prediction accuracy for four models along with 10, 20, 30, 40 and 50 principal components on DIP database. It can be found that, with the increase of principal components, the generalization performance is amazingly not always improved. It shows a decrease trend when the principal components exceed 20 for SVM, C_GSVM, SOM_GSVM and 10 for M_GSVM. This means that when using PCA to preprocess DIP database, many negative features will be deleted during classification. It is supported in Table 8 that the worst results are obtained when all of 1260 features are selected, hence, we may only need less principal components on solving practical problems.

6. Conclusions

In order to improve the efficiency of SVM, traditional GSVM models are usually trained after data mapping, simplification, granulation and other operations. However, the model error is inevitably and thus limits the improvement of generalization performance. This paper proposes a granular support vector machine model based on mixed measure, which granulates in high dimensional space and extracts some mixed granules for SVM training. The hyperplane will be further corrected by geometric analysis. The M_GSVM can retain the sufficient original support vector information, enhance the potential improvement of generalization performance, and reduce the model error effectively. By this model, high generalization performance can be obtained with high learning efficiency simultaneously.

Because this paper focuses on improving the generalization performance of GSVM in a given kernel space, the kernel selection and parameter tuning are not taken into account. How to combine M_GSVM model with kernel selection will be our future work. Additionally, how to set the model parameters to make M_GSVM method be applied to different types datasets is worthy to further exploration. Besides, the combination of the proposed M_GSVM

with effective feature reduction approaches to predict the proteins interaction is also a valuable problem.

Algorithm 1 Granule dividing algorithm

Step1: *Select k samples randomly as the center of k granules.*
Step2: *Classify samples according to formula (6) by nearest neighbor approach in kernel space.*
Step3: *Adjust the centers of k granules by* Eq. (4)*, and observe whether there are changes of these centers. If so, back to step2, else go to step4.*
Step4: *End the algorithm and obtain the divided granules* $\{X_1,X_2,\ldots,X_k\}$.

Algorithm 2 M_GSVM algorithm

Initialization
Given the training samples $X=\{(x_i,y_i)\}_{i=1}^{l}$.
Step1: Granulating based on kernel.
Given the number of granulation parameters k, and take granulation based on granular dividing algorithm. Obtain the divided granules $\{X_1,X_2,\ldots,X_k\}$.
Step2: Extracting mixed granules.
Step2.1: Set up the threshold parameter of mixed granule $support_i$ and $purity_i$, then take the samples of mixed granule into the *Set*(*mixed*).
Step2.2: while (the size of granule X_i in *Set*(*mixed*) is bigger than $2l/k$)
loop
{
Delete the mixed X_i from *Set*(*mixed*).
Divide mixed granule X_i into sub granules based on granular dividing algorithm.
Add mixed sub granules into the *Set*(*mixed*).
}
Step3: Training SVM
Take all the samples of *Set*(*mixed*) as training samples, and train SVM. Then an initial approximate hyperplane f is obtained.
Step4: Correcting hyperplane.
Step4.1: Set up the threshold parameter d' ($d'>d$) of hyperplane correction.
Step4.2: Compute the distance from each purity granule super ball to the initial hyperplane f according to Eq. (9).
Step4.3: For each purity granule X_i
{
Add all samples x_{ij} in granule X_i into the training dataset, if ($d(x_{ij},f)<d'$).
}
Step5: Retraining SVM
Train SVM on the new training samples and obtain the final superior hyperplane f.

Acknowledgement

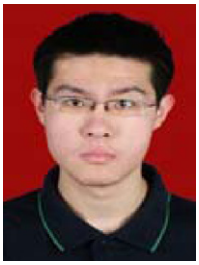
The work described in this paper was partially supported by the National Science Foundation of China (No. 60975035, 71031006, 61273291), Doctoral Fund of Ministry of Education of China (No.20091401110003), Key Project of Natural Science Foundation of Shan Xi Province (No.2009011017-2), Research Project Supported by Shanxi Scholarship Council of China (No. 2012-008), and Graduate Innovation Project of Shan Xi Province (No.20103021).

References

[1] V. Vapnik, Statistical Learning Theory, Wiley, New York, 1998 493–520.
[2] I.W. Tsang, J.T. Kwok, P.M. Cheung, Core vector machines: fast SVM training on very large datasets, J. Mach. Learn. Res. 6 (2005) 363–392.
[3] F.L. Cao, X. Xing, J.W. Zhao, Learning rates of support vector machine classifier for density level detection, Neurocomputing 82 (2012) 84–90.
[4] J.S. Nath, S.K. Shevade, An efficient clustering scheme using support vector methods, Pattern Recognit. 39 (8) (2006) 1473–1480.
[5] W.J. Wang, Z.B. Xu, A heuristic training for support vector regression, Neurocomputing 61 (2004) 259–275.
[6] X.M. Wang, F.L. Chung, S.T.W. On, minimum class locality preserving variance support vector machine, Pattern Recognit. 43 (8) (2010) 2753–2762.
[7] R. Wang, S. Kwong, D.G. Chen, Inconsistency-based active learning for support vector machines, Pattern Recognit. 45 (10) (2012) 3751–3767.
[8] W.J. Wang, C.Q. Men, V.Z. Lu, Online prediction model based on support vector machine, Neurocomputing 71 (4–6) (2008) 550–558.
[9] J.T. Yao, A ten year review of granular computing, in: Proceedings of 2007 IEEE International Conference on Granular Computing, 2007, pp. 734–739.
[10] Y.C. Tang, B. Jin, Y. Sun, Y.Q. Zhang, Granular support vector machines for medical binary classification problems, In: Proceedings of the IEEE CIBIB. Piscataway, HJ: IEEE Computational Intelligence Society, 2004, pp. 73–78.
[11] E. Osuna, R. Freund, F. Girosi, Training support vector machines: an application to face detection, IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (1997) 130–136.
[12] C.P. John, Fast training of support vector machines using sequential minimal optimization, in: B. Scholkopf, et al., (Eds.), Advances in Kernel Methods-Support Vector Learning, MIT Press, Cambridge, MA, 1999, pp. 185–208.
[13] C.C. Chang, C.J. Lin, LIBSVM-a library for support vector machines, From ⟨http://www.csie.ntu.edu.tw/∼cjlin/libsvm⟩, 2001.
[14] X.G. Zhang, Using class-center vectors to build support vector machines, Proc. Neural Net. Signal Process'99 (1999) 3–11.
[15] H. Yu, J. Yang, J.W. Han, X.L. Li, Making SVMs scalable to large datasets using hierarchical cluster indexing[J], Data Min. Knowl. Discovery 11 (3) (2005) 295–321.
[16] S.X. Cheng, Y.S. Frank, An improved incremental training algorithm for support vector machines using active query, Pattern Recognit. 40 (3) (2007) 964–971.
[17] H.S. Guo, W.J. Wang, C.Q. Men, A novel learning model-kernel granular support vector machine, In: Proceedings of 2009 International Conference on Machine Learning and Cybernetics, Baoding, China. IEEE Press, 2009, pp. 930–935.
[18] X.L. Tang, L. Zhuang, J. Cai, C.B.L, Multi-fault classification based on support vector machine trained by chaos particle swarm optimization, Knowledge Based Syst. 23 (5) (2010) 486–490.
[19] P.F. Pai, M.F. H, M.C. Wang, A support vector machine-based model for detecting top management fraud, Knowledge based Syst. 24 (2) (2011) 314–321.
[20] A.K. Deb, M. Jayadeva, S. Gopal, S.V.M. Chandra, based tree-type neural networks as a critic in adaptive critic designs for control, IEEE Trans. Neural Networks 18 (4) (2007) 1101–1114.
[21] Y.C. Tang, B. Jin, Y.Q. Zhang, Granular support vector machines with association rules mining for protein homology prediction, Artif. Intell. Med. 35 (1–2) (2005) 121–134.
[22] V. Vapnik, A. Chervonenkis, Necessary and sufficient conditions for the uniform convergence of means to their expectations, Theor. Probab. Applic. 26 (3) (1981) 532–553.
[23] V. Vapnik, A. Chervonenkis, The necessary and sufficient conditions for consistency in the empirical risk minimization method, Pattern Recognit Image Anal. 1 (3) (1991) 283–305.
[24] W.J. Wang, Z.B. Xu, V.Z. Lu, et al., Determination of the spread parameter in the Gaussian kernel for classification and regression, Neurocomputing 55 (3–4) (2003) 643–663.
[25] W.J. Wang, J.L. Guo, C.Q. Men, An approach for kernel selection based on data distribution, Lect. Notes Artif. Intell. 5009 (2008) 596–603.
[26] K.P. Wu, S.D. Wang, Choosing the kernel parameters for support vector machines by the inter-cluster distance in the feature space, Pattern Recognit. 42 (5) (2009) 710–717.
[27] P. Zhou, D.H. Li, H. Wu, F. Cheng, The automatic model selection and variable kernel width for RBF neural networks, Neurocomputing 74 (17) (2011) 3628–3637.
[28] S.S. Zhou, H.W. Liu, Y. Feng, Variant of Gaussian kernel and parameter setting method for nonlinear SVM, Neurocomputing 72 (13–15) (2009) 2931–2937.
[29] P. Wittek, C.L. Tan, Compactly supported basis functions as support vector kernels for classification, IEEE Trans. Pattern Anal. Mach. Intell. 33 (10) (2011) 2039–2050.
[30] UCI Machine Learning Repository, 2010. Available from: ⟨http://ww.ics.uci.edu/mleam/MLRepository.html⟩.

Wang Wenjian, received the B.S. degree in computer science from Shanxi University, China, in 1990, the M.S. degree in computer science from Hebei Polytechnic University, China, in 1993, and Ph.D. degree in applied mathematics from Xi'an Jiao Tong University, China, in 2004. She worked as a research assistant at the Department of Building and Construction, the City University of Hong Kong from May 2001 to May 2002. She has been with the Department of Computer Science at Shanxi University since 1993, where she was promoted as Associate Professor in 2000 and as Full Professor in 2004, and now serves as a Ph.D supervisor in Computer Application Technology and System Engineering. She has published more than 70 academic papers on machine learning, computational intelligence, and data mining. Her current research interests include neural networks, support vector machines, machine learning theory and environmental computations et al.

Guo Husheng is a Ph.D. candidate from the School of Computer and Information Technology at Shanxi University, China. He received his B.S. and M.S. degrees from Shanxi University in 2008 and 2010, respectively. His research interests include support vector machines, machine learning and data mining.



Jia Yuanfeng is a M.S. candidate from the School of Computer and Information Technology at Shanxi University, China. He received his B.S. degree from Shanxi University in 2009. His research interests include machine learning and bioinformatics.

Bi Jingye is a M.S. candidate from the School of Computer and Information Technology at Shanxi University, China. He received his B.S. degree from Shanxi University in 2009. His research interests include data analysis and bioinformatics.