Applied Mathematics, Modeling and Computer Simulation
C. Chen (Ed.)

doi:10.3233/ATDE220030

Large-Scale Support Vector Machine Classification Algorithm Based on Granulation Mechanism

Yunsheng SONG[1], Xiaohan KONG, Rui DING, Zhenbin LIU, Zhaokun HUANG
College of Information Science and Engineering, Shandong Agricultural University, China

Abstract. Support vector machine classification algorithm has been deeply studied in the field of intelligent information processing with its solid theoretical foundation and excellent classification performance, and it has been widely used in real life, such as face recognition, text classification, sentiment analysis, and spam filtering, etc. The time complexity of the classic support vector machine algorithm is proportional to the square of the data size, and this poses a serious challenge to the calculability, effectiveness and efficiency of the support vector machine algorithm for massive data. In this paper, a large-scale support vector machine classification algorithm based on the granulation mechanism is proposed from the granulation and fusion of data information. A data granulation mechanism is firstly constructed using binary tree and classification label information, as well as the core sample set is obtained by combining with instance selection method, and then the core sample set and the remaining samples construct multiple weak support vector classifiers; finally draw on the idea of granularity fusion to generate a strong classifier. The experimental results on the standard data set verify the effectiveness and efficiency of the proposed algorithm, and it provides new research ideas and processing methods for processing massive data.

Keywords. Granular computing, granulation mechanism, support vector machine.

1. Introduction

Granular computing is a discipline that specializes in the study of thinking mode, problem solving, information processing mode based on granular structure-based thinking, it is a new computing paradigm in the current field of intelligent information processing [1,2]. From the perspective of data processing, granular computing granulates massive data and replaces samples with information granules as the basic processing unit to achieve efficient calculations. From the complex structure hidden in the massive labeled data to the performance requirements of efficient supervised classification algorithms, the high efficiency requirements of supervised classification algorithm modeling are highly consistent with the computational paradigm of granular computing.

[1] Corresponding Author, Yunsheng SONG, College of Information Science and Engineering, Shandong Agricultural University, China; Email: sys_sd@126.com.

Support vector machine (SVM) is one of the most representative algorithms in supervised learning, which is to find a linear classifier with the margin maximization in the feature space. Due to the relatively few related parameters and high generalization performance, SVM is widely used in practice. However, the process of constructing an SVM classifier is actually to solve a quadratic optimization problem (QP), and its time complexity is proportional to N^3, where T is the number of examples in the training set T. Therefore, the classic SVM algorithm is difficult to implement in an effective time in the face of large-scale data. For this reason, a large number of experts and scholars have begun to improve the training efficiency of the SVM algorithm from the perspective of algorithm optimization and instance selection [3]. The former mainly accelerates the SVM training process by decomposing the original optimization problem into several sub-optimization problems using data partition; the latter aims to look for one subset of with all the boundary instances as much as possible, and then uses this subset to replace the original training set to train the SVM classifier.

In terms of algorithm optimization. Osuna et al. [4] proposed a new type of QP decomposition algorithm using the idea of set partitioning, which completely avoided the assumption of the number of support vectors. Chang et al. proposed a parallel SVM algorithm using row-based approximate matrix decomposition. Compared with the original algorithm, its time complexity is greatly reduced $O\left(I\left(Np/N\right)\right)$, where p,m is the number of rows of the decomposed matrix and the number of machines. At the same time, Hush et al. [5] pointed out that the time complexity of most existing decomposition QP algorithms is $O\left(I\left(Np+q^3\right)\right)$ where I,q is the number of iterations and the size of the work set, and increases with the increase of. In addition, empirical research has shown that the time complexity of the ordinary decomposition QP algorithm is proportional to $O\left(N^q\right)$ $O(N^q)$, where q varies from 1.7 to 3.0 due to different problems. To this end, Singh et al. [6] proposed a distributed boundary-preserving kernel SVM algorithm (QP-SVM) for big data, which uses K-means clustering algorithm to divide each class sample into equal the number of subsets, and then divide the divided subsets into a new subset according to the consistency criterion. The experimental results on the standard data set show that the QP-SVM algorithm has higher execution efficiency than the typical distributed SVM algorithm. However, due to the limitation of the algorithm itself and the need for a large amount of domain knowledge and professional skills, it is very difficult to simply improve the performance of the SVM algorithm [7].

The hyperplane training mainly relies on support vectors in SVM algorithm, and support vectors are often the instances close to the classification boundary. Therefore, the training time of the classifier will be greatly reduced if these critical instances are obtained. Koggalage [8] proposed a new method of selecting core instances on the basis of Almeida's work. For the homogenous subsets, the instances located at the edges of it are selected as key instances instead of the subset center. However, these cluster-based instance selection algorithms have common shortcomings: the selected instance is completely dependent on the results of the clustering algorithm, and these results may be unstable. For the shortcomings of the cluster-based instance selection algorithm, Shin et al. [9] proposed the NPPS algorithm based on the label difference between the instance and its neighbors. Related experiments show that the SVM classifier trained on the subset

obtained by the NPPS algorithm obtains approximately the same test accuracy as that on the entire training set, but its training Time is greatly reduced. Angiul [10] proposed a fast compression close proximity algorithm (FCNN), which obtains a subset of instances that meets the consistency, that is, the label of each instance in the training set is the same as the label of its nearest neighbor. In addition, There are a large number of literatures that use the idea of nearest neighbors to propose a large number of instance selection algorithms, but such algorithms need to select appropriate kernel parameters to obtain results similar to or better than the original SVM algorithm[11].

This paper uses the idea of ensemble learning to propose a support vector machine algorithm (DP-SVM) that combines data partitioning and instance selection. First, the selection algorithm is used to obtain the core instance subset of the training set, and then the remaining instance subsets randomly divide into several subsets and merge them into new subsets respectively. Finally, train the SVM classifier on the subsets and use the voting principle to provide labels for the instances to be identified. The experiments on the real data set show that it is different from the original Compared with the SVM algorithm, the training efficiency of the DP-SVM algorithm is greatly improved while the price is adjusted to ensure the classification accuracy.

2. Main Work

Let $T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$ be the training set of examples from c different classes, $x_i = (x_{i1}, x_{i2}, \cdots, x_{im})$ is the i-th instance denoted by m attributes with its label $y_i \in (\omega_1, \omega_2, \cdots, \omega_c)$, and N is the number of instances.

DP-SVM algorithm is one kind of ensemble learning, and the classifier on based data partition and instance selection. It first uses instance selection algorithm to obtain a subset of core instances S, and then divides the training set $T - S$ into n subsets TS_i and merges them into a new subset $T_i = S \bigcup TS_i$, finally train the SVM classifier $f_i(x)$ on each subset and use the voting principle to construct the classifier among them.

There exist two important factors in ensemble learning, the difference among sub-classifiers and the high classification performance of sub-classifiers. Though the training process of SVM algorithm needs all the training instances, a small number of instances as support vectors are used to generate the classifier. Instance selection algorithms could search the critical instances to train the classifier of the similar classification performance with the classifier using all the training instances. The new subset T_i includes the core set S obtained by instance selection algorithm, and it has much more instances from the subset $T - S$. Therefore, the sub-classifiers trained on T_i could also obtain a good classification performance. On other hand, SVM algorithm is an unstable algorithm according to Breiman's conclusion on the stability of learning algorithms, a small change on training instances could take a large change on SVM classifier [12]. Besides the same instances in the core instance set S , the elements of the set $T - S$ in each divided subset $T_i (i = 1, 2, \cdots, n)$ are different because they are randomly selected from the set

$T-S$. Combining the instability of SVM algorithm and the different instances in each divided subset T_i, the sub-classifiers trained on T_i have a large difference. In conclusion, DP-SVM algorithm could achieve a good classification performance based on the above analysis.

The size of the divided subset T_i has a significant impact on the performance of the classifier $f_i(x)$ trained on it in DP-SVM algorithm. There are much fewer critical instances than others in most of the real datasets, and instance selection algorithms usually can obtain a small set of core instances whose size is not determined in advance. Then the size of the subset T_i is affected by the parameter $|T-S|/n$. The paper [13] suggests $n=[|T-S|^{0.3}]$ for different dataset, where $[|T-S|^{0.3}]$ is the minimum integer bigger the $|T-S|^{0.3}$. Finally, the pseudo-code of DP-SVM algorithm is listed in Algorithm 1.

Algorithm 1: DP-SVM algorithm

Input : Dataset $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$, the parameter $\theta=0.3$.

Output: The final classifier $f(x)$.

Adjust the number of subsets $n=[|T-S|^{\theta}]$;

Obtain the core instance set $S\subseteq T$;

Randomly divide the set $T-S$ into n subsets $TS_1,TS_2,\cdots,TS_n$;

for *l=1 to n* **do**

  Train SVM classifier $f_i(x)$ on the set $T_i=S\bigcup TS_i$;

end

$$f(x)=\arg\max_{j=1,2,\cdots,c}\sum_{i=1}^{n}I_{\{f_i(x)=\omega_j\}};$$

Return $f(x)$

Compared with the original SVM algorithm, the time complexity of the DP-SVM algorithm is proportional to $n(|T_i|)^3$, and its running time is greatly reduced, where $|T_i|$ is the size of the subset. In addition, DP-SVM algorithm has another advantage, which can utilize parallel computing. The classifiers are trained independently on the training of each instance subset, so each classifier can be trained at the same time. On the other hand, there is very little interaction between nodes during parallel computing, it just combines several classifiers into the final classifier.

Instance selection algorithm as an important data preprocessing method is widely used in the field of data mining, where FCNN algorithm is chosen in this paper for its advantages of lower time complexity and high compression rate [10]. However, FCNN algorithm needs to traverse all data multiple times, so it is difficult to efficiently process large-scale data. To this end, we adopt a divide-and-conquer strategy to speed up its operation. First, the original data is divided into several subsets, and then FCNN algorithm is run independently on each subset, and finally the results on all the subsets

are merged into the final result. Although the time complexity of this algorithm is not reduced, its running time is greatly reduced. In addition, since the FCNN algorithm runs independently on each subset, parallel computing can be used.

Algorithm 2: Instance selection based on data partition

Input : Dataset $T=\left\{\left(x_1,y_1\right),\left(x_2,y_2\right),\cdots,\left(x_N,y_N\right)\right\}$, the divided subset size *s*.

Output: The subset $S\subset T$.

$d=[\log_2\left(N^{\beta}\right)],T_i=S=\varnothing,\ i=1,2,\cdots,2^d$;

for *l*=1 to *c* **do**

  $T_{0,1}^l=\left\{\left(x_i,y_i\right)\in T:y_i=\omega_l\right\}$

  for *i*=0 to *d*-1 **do**

    Randomly sample a feature from all the features without replacement;

    for *j*=1 to 2^i **do**

      Divide the set $T_{i,j}^l$ into two subsets $T_{i+1,2*(j-1)+1}^l$ and $T_{i+1,2*j}^l$ using the median of the sampled feature within $T_{i,j}^l$;

    end

  end

end

for i=1 to 2^d **do**

  for l=1 to c **do**

    Random sample a subset T_0 from the set $\left\{T_{ws,j}^l:1\le j\le 2^{ws}\right\}$ without replacement, and $T_i=T_i\bigcup T_0$;

  end

  Run FCNN algorithm on T_i to get the subset S_i and $S=S_i\bigcup S$;

end

2 Return *S*

The binary tree is a special data structure that uses different attributes of data to recursively divide the input space into several regions. If the instances in each region are regarded as a subset of instances, the binary tree can be regarded as a division of data. In order to speed up the process of building a binary tree, the attribute is randomly selected as the splitting attribute, and the median of the value of the attribute in the current instance is selected as the splitting point. Starting from the root node, all current instances are divided into two subsets of approximately equal size each time, so the number of instances in the final leaf node is also approximately equal. If the number of levels of this binary tree is *d*, the number of subsets after the final division is 2^d, and the size of each subset is approximate $[N/2^d]$. Under the condition of the fixed subset size *s*, only the number of layers needs to be adjusted, i.e. $d=[\log_2(N/s)]$. However, this algorithm is not directly applied to the training set, because all instances in a certain subset or some subsets may be from the same category, which does not meet the requirement of heterogeneous subsets. To this end, we adopt a hierarchical division method. Each set of examples from the same class is firstly divided into 2^d disjoint

subsets. It successively and randomly selects one subset from the divided subsets of each class, and merges these selected subsets into a new set, this process repeats 2^d times to get 2^d different divided subset of the training set. Based on the above content, the implementation details of the instance selection algorithm based on data partition are listed in algorithm 2.

3. Experimental Analysis

3.1. Experiment Setup

In order to evaluate the performance of DP-SVM, three current representative algorithms of LIBSVM algorithm [14], DC-SVM algorithm [15] and DIP-SVM algorithm [16] are selected as the objects to be compared. Ten standard data sets are selected from the LIBSVM database, and the capacity of each data set is greater than 10,000. All the data set information of the experiment is listed in table 1.

Table 1. Ten data sets used in the experiment

Dataset	Size	Feature	Class
Acoustic	98528	50	3
Cod-rna	488565	8	2
Combined	98528	100	3
Covtype	581012	54	7
ijcnn1	141691	22	2
MiniBooNE	130065	50	2
Seismic	98528	50	3
Sensorless	58509	48	11
Skin	245057	3	2
Susy	5000000	18	2

Three state-of-the-art SVM acceleration algorithms are selected to be compared: divide and conquer SVM with multiple levels (DC-SVM), distribution preserving kernel support vector machine (DIP-SVM), a library for support vector machines (LIBSVM). The literature provides a large number of indicators for evaluating the performance of classification algorithms, among which classification accuracy (Acc), Kappa coefficient (Kappa) and training time are the three most commonly used indicators. The first two reflect the generalization performance of the classifier, while the latter reflects the speed of the algorithm. We choose the fold cross-validation method to effectively estimate the values of the three evaluation indicators. In this method, first randomly divide the current data into subsets, select a subset as the training set and the remaining subset as the test set, and then repeat this selection, and finally use the average of the results as the final result. In order to obtain the experimental results under the same conditions, we run all the experiments on the same machine, and all the programs are written in MATLAB, and the SVM algorithm is called the LibSVM algorithm written in C language. According to Delgado's conclusion on the choice of the kernel function in the literature [17], the Gaussian kernel function is selected, and the default parameters are all selected for the

kernel parameters. Furthermore, Wilcoxon sign-rank test [18] is used to test whether there exists a significant difference between the proposed algorithm and another algorithm. In all the following experiments, $\theta = 0.7$, the significance level is 0.05, and k=10.

3.2. Classification Performance

This section uses classification accuracy and Kappa coefficient to evaluate the generalization performance of the four algorithms. Table 2 lists the test accuracy and Kappa coefficient of the two algorithms on different data sets.

Table 2. Acc and Kappa of different algorithms

Dataset	LIBSVM		DC-SVM		DIP-SVM		DP-SVM	
	Acc	Kappa	Acc	Kappa	Acc	Kappa	Acc	Kappa
Acoustic	0.770	0.634	0.732	0.582	0.742	0.592	0.747	0.593
Cod-rna	0.963	0.917	0.923	0.818	0.915	0.886	0.960	0.910
Combined	0.865	0.787	0.853	0.767	0.848	0.752	0.859	0.778
Covtype	0.806	0.682	0.842	0.685	0.843	0.698	0.771	0.627
Ijcnn1	0.969	0.794	0.970	0.803	0.958	0.673	0.949	0.638
MiniBooE	0.855	0.692	0.836	0.656	0.864	0.742	0.840	0.652
Seismic	0.759	0.619	0.682	0.512	0.693	0.515	0.692	0.528
Sensorless	0.947	0.942	0.910	0.932	0.923	0.913	0.887	0.875
Skin	0.994	0.981	0.993	0.979	0.994	0.984	0.985	0.957
Susy	0.796	0.583	0.800	0.595	0.793	0.584	0.802	0.595
Mean	0.872	0.763	0.854	0.733	0.857	0.734	0.849	0.715
Median	0.860	0.740	0.848	0.726	0.856	0.720	0.850	0.645

It can be seen from table 2 that the classification accuracy and Kappa coefficient of the DC-SVM algorithm, DIP-SVM algorithm and DP-SVM algorithm on all data sets are slightly lower than the LIBSVM algorithm, because the first three algorithms are based on LIBSVM algorithm to perform related operations on the data for achieving the purpose of improving the training efficiency. However, on the whole, the average difference between the four algorithms on all data sets is relatively small. Among them, the average classification accuracy of all algorithms on all data sets are 0.872, 0.854, 0.857, 0.849, the average Kappa coefficient is 0.763, 0.733, 0.734, 0.715; the median of the classification accuracy is 0.860, 0.848, 0.856, 0.850, its median value of Kappa coefficient is 0.710, 0.726, 0.720, 0.645.

The Wilcoxon signed rank is adopted test further study whether there is a difference in test performance between the proposed algorithm and another algorithm. For the classification accuracy index, the p-values between the proposed algorithm and the DC-SVM algorithm and the DIP-SVM algorithm are 0.922 and 0.557, respectively; in the Kappa coefficient index, the p-values are 0.551 and 0.232, both of which are greater than the given significance level 0.05. Therefore, there is no significant difference in the generalization performance of the three acceleration algorithms.

3.3. Training Efficiency

The learning algorithm needs to obtain a classifier with strong generalization performance in a limited time for the large-scale data. To effectively evaluate the training efficiency of the acceleration algorithm, the acceleration ratio is selected as the metric,

that is, the ratio of the training time of the LIBSVM algorithm to the training time of the acceleration algorithm. The larger value of the acceleration ratio, the higher the training efficiency of the algorithm. Table 3 lists the acceleration ratio of the three acceleration algorithms on different data sets.

Table 3. Acceleration ration of different algorithms

Dataset	DC-SVM	DIP-SVM	DP-SVM
Acoustic	2.341	2.562	2.717
Cod-rna	29.922	125.635	146.429
Combined	4.323	5.146	5.328
Covtype	10.643	48.242	63.257
Ijcnn1	5.24	45.691	62.324
MiniBooNE	12.235	55.982	42.325
Seismic	7.626	9.823	12.044
Sensorless	2.254	3.453	5.935
Skin	6.289	42.231	45.585
Susy	1.334	1.965	4.962

It can be seen from table 3 that on all data sets except the MiniBooNE data set, the training efficiency of the DP-SVM algorithm is significantly higher than that of DC-SVM and DiP-SVM algorithms. In terms of average conditions, the average training acceleration ratios of the three algorithms on all data sets are 8.221, 34.073 and 39.077, respectively. Because the capacity of each data set is different, and the training efficiency of each algorithm is more sensitive to the capacity of the data set. For this reason, Wilcoxon signed rank test is also used to test whether there is a significant difference between the training speedups of the three algorithms. The test p-value between the DP-SVM algorithm and the DC-SVM algorithm and the DIP-SVM algorithm are 0.002 and 0.037, both of which are less than the given significance level of 0.05. Therefore, the DP-SVM algorithm has the highest training efficiency.

4. Conclusion

The proposed algorithm provides an effective method for granulating large-scale data, which can effectively ensure the difference between the sample label and the sample size of each data granule. Uniformity, and finally realize the selection of effective and efficient core instances. On this basis, combined with granular fusion, a large-scale SVM acceleration mechanism is given. The experimental results on the standard data set show that the algorithm proposed in this paper is trained under the condition of ensuring accuracy. The efficiency is greatly improved. This achievement has important theoretical significance and application value for intelligent information processing in a massive data environment. In the future, we will study how to adaptively determine the relevant parameters in the algorithm for different datasets.

Acknowledgments

This work was also supported by Major Scientific and Technological Innovation Project of Shandong Province (No.2019JZZY010716), Project ZR2020MF146 supported by Shandong Provincial Natural Science Foundation, Open Project Foundation of Intelligent Information Processing Key Laboratory of Shanxi Province (CICIP2021002).

References

[1] Guo S and Zhao H 2021 *Artificial Intelligence Review* 54 2067–2089
[2] Yao Y 2020 *International Journal of Approximate Reasoning* 116 106–125
[3] Cheng F, Chen J, Qiu J and Zhang L 2020 *Neurocomputing* 394 70–83
[4] Osuna E, Freund R and Girosit F 1997 *Proceedings of IEEE computer society conference on computer vision and pattern recognition* pp 130–136
[5] Hsieh C J, Chang K W, Lin C J, Keerthi S S and Sundararajan S 2008 *Proceedings of the 25th international conference on Machine learning* pp 408–415
[6] Hush D, Kelly P, Scovel C, Steinwart I and Sch¨olkopf B 2006 *Journal of Machine Learning Research* 7
[7] Chauhan V K, Dahiya K and Sharma A 2019 *Artificial Intelligence Review* 52 803–855
[8] Koggalage R and Halgamuge S 2004 *Neural Information Processing-Letters and Reviews* 2 57–65
[9] Shin H and Cho S 2007 *Neural Computation* 19 816–855
[10] Angiulli F 2007 *IEEE Transactions on Knowledge and Data Engineering* 19 1450–1464
[11] Tang T, Chen S, Zhao M, Huang W and Luo J 2019 *Soft Computing* 23 3793–3801
[12] Breiman L 1996 *The annals of statistics* 24 2350–2383
[13] Kleiner A, Talwalkar A, Sarkar P and Jordan M I 2014 *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 76 795–816
[14] Chang C C and Lin C J 2011 *ACM Transactions on Intelligent Systems and Technology* 2 27
[15] Hsieh C J, Si S and Dhillon I S 2014 *Proceedings of the 31th International Conference on Machine Learning* pp 566–574
[16] Singh D, Roy D and Mohan C K 2017 *IEEE Transactions on Big Data* 3 79–90
[17] Fern´andez-Delgado M, Cernadas E, Barro S and Amorim D 2014 *The Journal of Machine Learning Research* 15 3133–3181
[18] Dem˘sar J 2006 *The Journal of Machine Learning Research* 7 1–30