The Third International Symposium on Optimization and Systems Biology (OSB'09)
Zhangjiajie, China, September 20–22, 2009
Copyright © 2009 ORSC & APORC, pp. 117–122

Density Clustering Based SVM and Its Application to Polyadenylation Signals*

Yuanhai Shao   Yining Feng   Jing Chen
Naiyang Deng†

College of Science, China Agricultural University, Beijing 100083, China

Abstract Support vector machines (SVM) have been promising methods for classification analysis due to their solid mathematical foundations. Clustering-based SVMs are used to solve large samples classification problems and reduce the computational cost. In this paper, we present a density clustering based SVM(DCB-SVM) method to predict polyadenylation signal (PAS) in human DNA and mRNA sequences. We decrease the original data scale by using the density restricted hierarchical clustering. This strategy leads to solving smaller sized problems, making DCB-SVM work faster than standard SVM. According to the results of the PAS experiment, the proposed method is not only fast, but also shows better improvement in sensitivity than the SVM.

Keywords Support vector machines; Polyadenylation signals; BIRCH algorithm

1 Introduction

Due to the rapid growth of the biological data, automatic methods for categorizing the DNA/RNA data are needed. Machine learning has been widely applied to bioinformatics and has gained a lot of success in this research area. With the increase of biological data, fast and efficient algorithms are very necessary.

Polyadenylation signal (PAS) prediction is an important problem in the bioinformatics [1]. The polyadenylation process is illustrated in [1, 2]. Prediction of poly(A) sites has been attempted by several groups during the last several years. Researchers have proposed some efficient methods to predict the PAS. Tabaska and Zhang [3] developed Polyadq, which employed two quadratic discriminant functions for sequences containing AAUAAA and AUUAAA. They also used a position weight matrix for the downstream sequence, a weighted average of hit positions for DSE, and downstream dimer preferences. In addition, weight-matrix-only [4] also has been employed for poly(A) site prediction. Liu et al. [5] first predicted the human PAS by using support vector machine(SVM) and got higher accuracy than the results from Polyadq and Erpind. While facing the large scale DNA sequences, SVM was restricted by the space and time cost. For large scale problems, it is necessary to reduce the data scale. Clustering based SVM can be used to

*This work is supported by the Key Project of the National Natural Science Foundation of China (No.10631070) and the National Natural Science Foundation of China (No.10601064).
†Corresponding Author. E-mail: dengnaiyang@vip.163.com

classify very large data sets with relatively low dimensionality, such as streaming data or data in large data warehouses.

From the idea of reducing the training data scale, we proposed the density clustering based SVM (DCB-SVM) in this paper. By limiting the number of training samples in each leaf node, density hierarchical clustering reflects the original distribution of the data set. Then we use the clustering center to denote the micro-cluster. At last, SVM is used to get the optimal decision function. According to the results of the PAS experiment by using human DNA sequence pattern, we get higher predicting precision in PAS data and achieve moderate sensitivity and specificity than clustering-based SVM(CB-SVM) [7].

2 Methods

2.1 Density restricted hierarchical clustering algorithm

In this section, we present the density restricted hierarchical micro-clustering algorithm, which is similar in spirit with BIRCH algorithm [6, 7]. The BIRCH algorithm builds a dendrogram called clustering feature tree (CF tree) while scanning the data set. The CF tree carries spherical shapes of hierarchical clusters and captures the statistical summaries of the entire data set. We construct a similar structure called density restricted clustering feature tree (DCF tree) for clustering. DCF tree also assigns an actual object in a cluster as the cluster center to facilitate it by clustering in any distance space, but DCF tree controls the number of the points in the nodes.

The concept of the CF tree is the core of the hierarchical micro-clustering algorithm which makes the clustering increase without expensive computations. Given N d-dimensional data points in a cluster : $\{x_i\}$, where $i = 1, \cdots, N$, the centroid C and radius R of the cluster are defined as:

$$C = \frac{\sum_{i=1}^{N} x_i}{N}, \tag{1}$$

$$R = (\frac{\sum_{i=1}^{N} \|x_i - C\|}{N})^{\frac{1}{2}}. \tag{2}$$

The CF vector of the cluster is defined as a triple : $CF = (N, LS, SS)$, where n is the number of data points in the cluster, LS is the linear sum of the n data points, and SS is the component-wise square sum of the n data points. CF tree is a height-balanced tree with two parameters: branching factor b and threshold t. The tree size is a function of t. The larger t is, the smaller the tree is. The branching factor b can be determined by memory page size such that a leaf or a non-leaf node fits in a page. The CF tree is a compact representation of the data set, because each entry in a leaf node is not a single data point but a cluster, which absorbs many data points within a radius of t or less.

In DCF tree, besides the triple clustering feature (N, LS, SS), we introduce a new parameter D in the clustering features tree, which is called the density parameter binding. D is used to limit the largest number of sample points in leaf node. It plays the similar role in branch parameter b, but b plays the bound role in the largest number of non-leaf node from the data storage and computer I/O operations, and the density parameter D is the role of restraint to limit each micro-cluster represented by the number of points of the original samples.

The density restricted hierarchical clustering tree is built up dynamically as new data objects inserted, the insertion is similar to the CF tree. The sketch is given below:

1. *Choosing the appropriate leaf*: Starting from the root, it traverses the CF tree down to the leaf level by choosing the child node whose centroid is closest to the new data object at each level.

2. *Modifying the leaf*: If the leaf nodes is too big to fit in memory, the leaf nodes will be split into two. The leaf contains the original data points of the database. If it conflicts with the parameters D of the nodes of all sample points, select two points whose distance is the farthest as the new leaf node of the seeds, and insert other points into the nearest one from the leaf nodes. Then update the clustering feature vectors, until all samples are assigned.

3. *Modifying the path*: Check the father nodes until the root node meets the parameter threshold of the branch. If unsatisfied, split it and recursively traverse back up to the root while performing the same checks. If the root node is unsatisfied, increase the height of the tree. Update cluster feature information on the path of every non-leaf node. If there is no split, just update the parameters.

2.2 Density clustering based SVM

The clustering algorithms based on the distance often tend to cluster in a circular area of some clusters, but the distribution of data is usually extremely complex. While clustering a non-circular on the regional distribution of the data sets by clustering algorithms based on the distance, the cluster centers do not properly reflect the distribution of the original data. Density restricted hierarchical clustering is expected to bound the various micro-cluster in the number of samples by the limitations of each micro-cluster in the density of sample points. If the micro-cluster data are too many that they tend to be a group of non-circular distribution of data to a circular cluster, the micro-cluster will be split by the restrictions of the density parameters.

Table 1. The DCB-SVM algorithm.

```
Input : The positive data set $P$, and the negative data set $N$.
Output : A boundary function of $f$.
Process :
  1. Density Restricted Hierarchical Clustering: Construct a positive and a
     negative tree from P and N respectively.
  2. Put the positive and negative root entries in initial training set S.
  3.1. $f' = SVM.train(S)$;     // construct a boundary $f$;
  3.2. $S' = getMargin(f, S)$; // compute the low margin entries $S'$ from $S$ using $f'$;
  3.3. $S = S - S'$;              // exclude the low margin data from $S$;
  3.4. $g = SVM.train(S)$;     // construct a boundary $f$;
  4. return $f$.
```

The key idea of DCB-SVM can be viewed as similar to that of selective sampling, i.e., selecting the data that maximizes the benefit of learning. We cluster the node entries near the boundary to get finer sampling close to the boundary and coarser sampling far from the boundary. Based on this idea, we use the SVs, the description of the class boundary, while keeping the total number of training data points as small as possible. In practice, soft constraints are usually necessary to cope with noise in the training set. Using soft constraints generates the SVs with different distances from the boundary.

DCB-SVM runs on the DCF tree, which can be constructed in a single scan of the entire data set. It carries the statistical summaries that facilitate efficient and effective construction of an SVM boundary. The sketch of the DCB-SVM algorithm is shown in Table1.

3 Implementation

3.1 Datasets

We use the sequence data provided by Legendre and Gautheret [4] and aim to predict the polyadenylation signal (PAS) in human sequences. The data set contains one group of training data (2327 true PAS) and 5 groups of testing data. Each of them consists of 982 samples. Among these 5 sets of testing data, one is true PAS and the other four are all false PAS. The data set can be downloaded from http:// tagc.univ-mrs.fr/pub/erpin/. Every sequence contains 206 bases and has a PAS in the center.

There are many mature methods to select the most obvious feature. In this paper, we used the Enterpriser Miner module in SAS 9.1.3 to select the feature through R-square method. First of all, compute the R-square contribution, and then select the first ten features which have the largest R-square contribution. The features we selected in the order of descending R-square contribution are: UP_TGT, DOWN_A, UP_AG, DOWN_TGT, DOWN_GGC, UP_AAG, UP_A, DOWN_AG, DOWN_GAA, UP_GGC.

For prediction, we used the following equations for Sensitivity (SN), Specificity (SP), FPR, precision and accuracy: $SN = TP/(TP+FN)$, $SP = TNR = TN/(TN+FP)$, $FPR = FP/(TN+FP) = 1-TNR$, $precision = TP/(TP+FP)$, $accuracy = (TP+TN)/(TP+FP+TN+FN)$, where, TP is true positive, TN is true negative, FN is false negative and FP is false positive.

3.2 Experimental results

Numerical experiments are conducted in the PC (Pentium(R) 4 CPU 2.90GHz 1024M RAM) hardware environment and Windows XP/Matlab7.0 [8] and libsvm [9] software environment. In the proceeding of the experiments, we compare the random sample support vector machine (10% and 20% random sample), SVM [5] and CB-SVM [7] with the density clustering based SVM by the same training data set. Training time and accuracy are used to evaluate the model efficiency, and then we compare the model performance with the results of the [5] and the CB-SVM on the five testing set.

Table 2. The results of the 10-fold cross validation for training data.

Algorithm	Data	Time	SN%	SP%	Precision%	AC%
SVM (10%)	442	3s	47.73	38.58	40.58	38.13
SVM (20%)	884	7s	61.79	57.91	60.09	58.85
SVM	4418	65s	82.79	**79.91**	**80.09**	**80.30**
CB-SVM	801	31s	80.60	76.78	77.49	78.48
Ours	834	35s	**83.42**	74.84	79.80	78.91

The 10-fold cross validation result of the model which are trained by the random sampling SVM , CB-SVM and DCB-SVM are shown in Table 2. In our algorithm, the positive SVs are 229 and negative SVs 189, the first SVM training time is 7s, and the second SVM training time is 28s. When we use the centrical of the cluster as training data set, DCB-SVM includes more information in the same level of training data set scale than the random sampling SVM. DCB-SVM generally performs better than the 20% random sampling SVM. In most cases, it is better than CB-SVM, and almost the same with the SVM.

Table 3. Compared accuracy of the programs for the prediction in PAS.

Program	TP	FN	SN%
SVM (20%)	432	550	43.99
SVM	553	429	56.3
CB-SVM	548	434	55.8
Ours	757	225	**77.08**

From Table 2, when we use the SVM on all the training samples, it uses more training time than DCB-SVM , and the latter gets higher sensitivity and nearly the same accuracy as the former. Compared with the other two methods, we can get the result in a higher precision within longer time. It illustrates that DCB-SVM performs better than the random sampling SVM and CB-SVM in the aspect of sensitivity and accuracy.

We get the results of the polyadenylation signals predicting validation on the five testing data set through the 20% random sampling SVM, CB-SVM , SVM and DCB-SVM. Because each of the testing data set only has one kind of samples, we can evaluate the model performance through accuracy. The results are showed in Table 3 and Table 4.

Table 4. Negative predictions and accuracy of the programs for negative test data.

Data set	Programm	TN	FP	TNR/SP%	FPR%
CDS	SVM (20%)	663	319	67.52	32.48
	SVM	887	95	90.3	9.70
	CB-SVM	803	179	81.78	18.22
	Ours	863	119	**87.88**	12.12
Introns	SVM (20%)	567	415	57.74	42.26
	SVM	775	207	78.90	21.10
	CB-SVM	641	341	65.27	34.73
	Ours	725	257	**73.83**	26.17
Simple shuffling	SVM (20%)	465	517	47.35	52.65
	SVM	942	40	95.90	4.10
	CB-SVM	835	147	**85.03**	14.97
	Ours	802	180	81.67	18.33
Markov 1st order	SVM (20%)	446	536	45.42	54.58
	SVM	765	217	77.90	22.1
	CB-SVM	547	435	55.70	44.30
	Ours	585	397	**59.57**	40.43

From Table 3, we can see that the accuracy of DCB-SVM reaches 77.08% in the true polyadenylation signals data set, which is higher than the result in [5]. At the same time, CB-SVM gets 55.8%, which is almost the same as the SVM. Table 4 shows that the accuracies of DCB-SVM are higher than 20% random sampling SVM, but lower than the results in [5]. Compare with the CB-SVM, in most cases, it can get higher accuracies (the boldface to inform).

As a result, comparing the DCB-SVM with the random sampling SVM, CB-SVM and standard SVM, the former performs moderate results in the negative testing data set than the latter. In the true polyadenylation signals data set, the DCB-SVM get higher accuracy than other methods.

4 Conclusions

With the idea of reducing the training data scale, we reduce the original data scale by using density restrict hierarchical cluster. In the process of hierarchical cluster, we combine density restrict to the clustering algorithm, and propose the density clustering based SVM. The proposed method can reflect the original distribution in data sets. We predict the polyadenylation signals from the DNA sequence by using the density clustering based SVM , CB-SVM and standard SVM. The numerical results have shown that the density clustering based SVM is not only fast, but gets higher sensitivity and similar specificity to the CB-SVM.

References

[1] Cheng Y, Miura R M, and Tian B. Prediction of mRNA polyadenylation sites by support vector machine. Bioinformatics, 22(19): 2320-5, 2006.

[2] Proudfoot N. New perspectives on connecting messenger RNA 30 end formation to transcription. Curr.Opin.Cell Biol. 16, 272-278, 2004.

[3] Tabaska J E, and Zhang M Q. Detection of polyadenylation signals in human DNA sequences, Gene, 231: 77-86, 1999.

[4] Legendre M, and Gautheret D. Sequence determinants in human polyadenylation site selection, BMC Genomics, 4(1):7, 2003.

[5] Liu H Q, Han H, Li J Y, and Wong L. An in-silico method for prediction of polyadenylation signals in human sequences. Genome Inform, 14: 84-93, 2003.

[6] Zhang T, Ramakrishnan R, and Livny M. BIRCH: An Efficient Data Clustering Method for Very Large Databases [C]. In Proc. ACM SIGMOD Int. Conf. Management of Data (SIGMOD'96), 103-114, 1996.

[7] Yu Hwanjo, Yang Jiong, Han Jiawei, and Li Xiaoli. Making SVMs Scalable to Large Data Sets using Hierarchical Cluster Indexing. Data Mining and Knowledge Discovery, 11, 295-321, 2005.

[8] http://www.mathworks.com, 2007.

[9] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm