HK.NCCP
International Journal of Intelligent Information and Management Science
ISSN: 2307-0692 *Volume 2, Issue 6, August 2013*

Core Set Extreme Learning Machine

Shuxia Lu[1], Bin Liu[2], Caihong Jiao[1]
[1]Key Lab. In Machine Learning and Computational Intelligence
College of Mathematics and Computer Science, Hebei University
Baoding 071002, China

[2] School of Mechanical and Vehicular Engineering, Beijing Institute of Technology
Beijing 100081, China

Abstract: A core set extreme learning machine (CSELM) approach is proposed in order to deal with large datasets classification problem. In the first stage, the core set can be obtained efficiently by using the generalized core vector machine (GCVM) algorithm. For the second stage, the extreme learning machine (ELM) can be used to implement classification for much larger datasets. Experiments show that the CSELM has comparable performance with SVM and ELM implementations, but is faster on large datasets.

Keywords: Core vector machine; Extreme learning machine; Support vector machine; Core set

1. Introduction

How to effectively deal with large-scale data is a hot issue in current research. In recent years, a variety of approaches have been proposed in large datasets problem. These methods include: the extreme learning machine (ELM) [1, 2], ELM is a single hidden layer feed forward network where the input weights and the biases are chosen randomly and the output weights are calculated analytically; a generalized perceptron with margin [3], which can deal with the large datasets problem; a general piecewise linear classifier [4], which can solve the nonlinear separable problem without kernel; the geometric algorithms to large margin classifier based on affine hulls [5]; by chunking or decomposition methods, for example, the well-known sequential minimal optimization (SMO) algorithm [6]; sampling techniques for kernel methods [7]; the core vector machine (CVM) [8, 9], Tsang et al. proposed the core vector machine (CVM) by utilizing an approximation algorithm for the minimum enclosing ball (MEB) problem in computational geometry, the CVM algorithm achieves an asymptotictime complexity that is linear in N and a space complexity that is independent of N, where N is the size of the training patterns； maximum vector-angular margin core vector machine (MAMCVM) [10], by connecting the CVM method with MAMC such that the corresponding fast training on large datasets can be effectively achieved.

In this paper, we focus on the large datasets effective classification problem, a core set extreme learning machine (CSELM) approach is proposed. It consists of two stages. The first stage is to obtain the core set of the large training dataset by using the GCVM algorithm. In the second stage, the ELM algorithm is utilized to train on the obtained core set and yields a decision function for classifying testing patterns. Experiments on large classification datasets also demonstrated that the CSELM has comparable performance with SVM and ELM implementations, but is much faster and can handle much larger datasets.

The rest of this paper is organized as follows. Section 2 reviews the GCVM and ELM; and presents the CSELM approach. In Section 3, the experimental results on several datasets are reported. Some conclusions are finally given in Section 4.

2. Core Set Extreme Learning Machine （CSELM）

2.1. The Generalized Core Vector Machine (GCVM）

In this section, we first review the generalized core vector machine (The generalized CVM, GCVM) algorithm as proposed in [9]. The GCVM algorithm is much faster and can handle much larger datasets than existing SVM implementations. The generalized CVM algorithm can be used with any linear/nonlinear kernel and can also be applied to kernel methods such as SVR and the ranking SVM. Moreover, like the original CVM, its asymptotic time complexity is again linear in N and its space complexity is independent of N, where N is the size of the training patterns.

The GCVM utilizes an approximation algorithm for the center constrain minimum enclosing ball (CC-MEB) problem, which will be briefly introduced in Section 2.1.1.

2.1.1. Center Constrain Minimum Enclosing Ball (CC-MEB)

Suppose the training set is denoted by $S=\{x_i \mid x_i \in \mathbb{R}^n,\ \ i=1,\cdots,N\}$, the minimum enclosing ball of S (denoted $MEB(S)$) is the smallest ball that contains all the points in S. In this paper, we denote the ball with center $\mathbf{c}$ and radius R by $B(\mathbf{c},R)$. Also, the center and radius of a ball $B(\mathbf{c},R)$ are denoted by $\mathbf{c}_B$ and r_B, respectively. Given an $\varepsilon>0$, a ball $B(\mathbf{c},(1+\varepsilon)R)$ is an $(1+\varepsilon)$ -approximation of $MEB(S)$ if $R\le r_{MEB(S)}$ and $S\subset B(\mathbf{c},(1+\varepsilon)R)$. $\varphi: x_i \to \varphi(x_i)$ denotes the feature map associated with a given kernel k, and $B(\mathbf{c},R)$ is the desired MEB in the kernel-induced feature space Γ.

The MEB problem finds the smallest ball containing all $\varphi(x_i)\in S$ in the feature space. In this section, we first augment an extra $\delta_i \in R$ to each $\varphi(x_i)$, forming $\begin{bmatrix}\varphi(x_i)\\ \delta_i\end{bmatrix}$. Then, we find the MEB for these augmented points, while at the same time constraining the last coordinate of the ball's center to be zero (i.e., of the form $\begin{bmatrix}\mathbf{c}\\ 0\end{bmatrix}$). The primal form of the center constrain minimum enclosing ball (CC-MEB) problem can be formulated as

$$\begin{aligned}&\min \quad R^2\\ &s.t. \quad \left\|\varphi(x_i)-\mathbf{c}\right\|^2+\delta_i^2\le R^2,\quad i=1,\cdots,N.\end{aligned} \tag{1}$$

The corresponding dual of (1) is the following QP problem

$$\begin{aligned}&\max \quad \boldsymbol{\alpha}^T(diag(\mathbf{K})+\boldsymbol{\Delta})-\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha}\\ &s.t. \quad \boldsymbol{\alpha}^T\mathbf{1}=1,\quad \boldsymbol{\alpha}\ge\mathbf{0}.\end{aligned} \tag{2}$$

where $K=[k(x_i,x_j)]=[\varphi(x_i)^T\varphi(x_j)]$ is the corresponding kernel matrix, and

$$\boldsymbol{\Delta}=[\delta_1^2,\cdots,\delta_N^2]^T\ge\mathbf{0}. \tag{3}$$

From the optimal $\boldsymbol{\alpha}$ solution of (2), we can recover R and $\mathbf{c}$ as

$$R=\sqrt{\boldsymbol{\alpha}^T(diag(\mathbf{K})+\boldsymbol{\Delta})-\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha}} \tag{4}$$

$$\mathbf{c}=\sum_{i=1}^{N}\alpha_i\varphi(x_i). \tag{5}$$

The squared distance between the center $\begin{bmatrix}\mathbf{c}\\ 0\end{bmatrix}$ and any point $\begin{bmatrix}\varphi(x_l)\\ \delta_l\end{bmatrix}$

$$\left\|\varphi(x_l)-\mathbf{c}\right\|^2+\delta_l^2=\left\|\mathbf{c}\right\|^2-2(\mathbf{K}\boldsymbol{\alpha})_l+k_{ll}+\delta_l^2. \tag{6}$$

which does not depend explicitly on the feature map φ.

Because of the constraint $\boldsymbol{\alpha}^T\mathbf{1}=1$ in (2), an arbitrary multiple of $\boldsymbol{\alpha}^T\mathbf{1}$ can be added to the objective without affecting its solution. In other words, for an arbitrary $\eta\in\mathbb{R}$, (2) yields the same optimal as

$$\begin{aligned}&\max \quad \boldsymbol{\alpha}^T(diag(\mathbf{K})+\boldsymbol{\Delta}-\eta\mathbf{1})-\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha}\\ &s.t. \quad \boldsymbol{\alpha}^T\mathbf{1}=1,\quad \boldsymbol{\alpha}\ge\mathbf{0}.\end{aligned} \tag{7}$$

Hence, any QP problem of the form (7), with the condition (3), can also be regarded as a special MEB problem, called center constrained MEB, i.e. CC-MEB. As pointed out by Tsang et al., CC-MEB can be approximately solved with the asymptotic linear time complexity O(N) and its space complexity independent of N for large datasets by using the generalized core vector machine.

2.1.2. The GCVM Algorithm

The GCVM algorithm is shown in Algorithm 1. Here, the core set, the ball's center, and radius at the *t*th iteration are denoted by $S_t, \mathbf{c}_t$, and R_t respectively. The GCVM algorithm requires the input of a termination parameter ε.

The core set can be obtained by using CC-CVM.

Algorithm 1. GCVM

Step 1 Initialize ε, $t=0, S_t, \mathbf{c}_t, R_t$

Step 2 Update the core set: if there is no training pattern that falls outside the ball $B(\mathbf{c}_t,(1+\varepsilon)R_t)$ in the corresponding feature space, $S=S_t$.

Step 3 Find $\mathbf{z}$ such that it is the farthest away from $\mathbf{c}_t$ in the corresponding feature space and set $S_{t+1}=S_t\bigcup\{\mathbf{z}\}$

Step 4 Find the new MEB: $B(\mathbf{c}_{t+1},R_{t+1})$

Step 5 Set $t=t+1$, and go to step 2.

2.2. Extreme Learning Machine (ELM)

The extreme learning machine (ELM) is a single hidden layer feed forward network where the input weights and the biases are chosen randomly and the output weights are calculated analytically [1].

Let $\aleph=\{(\mathbf{x}_i,\mathbf{t}_i)\mid \mathbf{x}_i\in\mathbb{R}^n,\mathbf{t}_i\in\mathbb{R}^m, i=1,...,N\}$ be a sample set. Standard SLFNs with *L* hidden nodes with activation function $g(x)$ can approximate these *N* samples with zero error are modeled as

$$\sum_{i=1}^{L}\beta_i g(\mathbf{w}_i\cdot\mathbf{x}_j+b_i)=\mathbf{t}_j \quad j=1,...,N. \tag{8}$$

where $\mathbf{w}_i$ is the weight vector connecting the *i*th hidden node and the input nodes, β_i is the weight vector connecting the *i*th hidden node and the output nodes, and

b_i is the threshold of the ith hidden node, $\mathbf{w}_i \cdot \mathbf{x}_j$ denotes the inner product of $\mathbf{w}_i$ and $\mathbf{x}_j$. These equations can be written compactly as

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T}. \tag{9}$$

where

$$\mathbf{H}(\mathbf{w}_1,...,\mathbf{w}_L,b_1,...,b_L,\mathbf{x}_1,...,\mathbf{x}_N) = \begin{bmatrix} g(\mathbf{w}_1\cdot\mathbf{x}_1+b_1) & .... & g(\mathbf{w}_L\cdot\mathbf{x}_1+b_L) \\ .... & .... & .... \\ g(\mathbf{w}_1\cdot\mathbf{x}_N+b_1) & .... & g(\mathbf{w}_L\cdot\mathbf{x}_N+b_L) \end{bmatrix}_{N\times L} \tag{10}$$

$\boldsymbol{\beta} = [\beta_1,\cdots,\beta_L]^T \in \mathbb{R}^{L\times m}$ and $\mathbf{T} = [\mathbf{t}_1,\cdots,\mathbf{t}_N]^T \in \mathbb{R}^{N\times m}$.

where $\mathbf{H}$ is called the hidden layer output matrix of the neural network.

Unlike the traditional function approximation theories which require adjusting input weights and hidden layer biases, input weights and hidden layer biases of SLFNs can be randomly assigned if the activation functions in the hidden layer are infinitely differentiable. After the input weights and the hidden layer biases are chosen randomly, SLFNs can be simply considered as a linear system $\mathbf{H}\boldsymbol{\beta} = \mathbf{T}$ and the output weights (linking the hidden layer to the output layer) of SLFNs can be analytically determined through simple generalized inverse operation of the hidden layer output matrices.

$$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T}. \tag{11}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$.

If the number of hidden nodes is equal to the number of distinct training samples, matrix $\mathbf{H}$ is square and invertible when the input weight vectors and the hidden biases are randomly chosen, and SLFNs can approximate these training samples with zero error.

The ELM algorithm of SLFNs can be summarized as the following three steps.

Algorithm 2. ELM

Let $\aleph = \{(\mathbf{x}_i,\mathbf{t}_i) \mid \mathbf{x}_i \in \mathbb{R}^n, \mathbf{t}_i \in \mathbb{R}^m, i = 1,...,N\}$ be a given training set, activation function is $g(x)$, and hidden node number is L,

Step1: Randomly assign input weight $\mathbf{w}_i$ and bias b_i $(i = 1,...,L)$. For any weights and biases are randomly chosen from any intervals of $\mathbb{R}^n$ and $\mathbb{R}$, respectively, according to any continuous probability distribution in Matlab (using Matlab *rand* function).

Step 2: Compute the hidden layer output matrix of the network denoted by $\mathbf{H} \in \mathbb{R}^{N\times L}$.

Step 3: Calculate the output weight $\boldsymbol{\beta} \in \mathbb{R}^{L\times m}$.

$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T}$,

where $\mathbf{T} = [\mathbf{t}_1,...,\mathbf{t}_N]^T \in \mathbb{R}^{N\times m}$. $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$.

2.3. The CSELM Algorithm

We can now give a fast training algorithm for large data-sets which is called the core set extreme learning machine (CSELM). It consists of two stages. The first stage is to obtain the core set of the large training dataset by using GCVM. In the second stage, the ELM algorithm is utilized to train on the obtained core set and yields a decision function for classifying testing patterns. CSELM can be summarized as follows:

Algorithm 3. CSELM

Stage 1: Using GCVM to obtain the core set.

Step 1 Initialize ε, $t = 0, S_t, \mathbf{c}_t, R_t$

Step 2 Update the core set: if there is no training pattern that falls outside the ball $B(\mathbf{c}_t,(1+\varepsilon)R_t)$ in the corresponding feature space, go to step 6.

Step 3 Find $\mathbf{z}$ such that it is the farthest away from $\mathbf{c}_t$ in the corresponding feature space and set $S_{t+1} = S_t \bigcup \{\mathbf{z}\}$

Step 4 Find the new MEB: $B(\mathbf{c}_{t+1}, R_{t+1})$

Step 5 Set $t = t+1$, and go to step 2.

Stage 2: Using ELM to train the core set S_t.

Given activation function $g(x)$, and hidden node number L,

Step 6 Randomly assign input weight $\mathbf{w}_i$ and bias b_i $(i = 1,...,L)$. For any weights and biases are randomly chosen from any intervals of $\mathbb{R}^n$ and $\mathbb{R}$, respectively, according to any continuous probability distribution in Matlab (using Matlab *rand* function).

Step 7 Compute the hidden layer output matrix of the network denoted by $\mathbf{H} \in \mathbb{R}^{N\times L}$.

Step 8 Calculate the output weight $\boldsymbol{\beta} \in \mathbb{R}^{L\times m}$.

$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T}$,

where $\mathbf{T} = [\mathbf{t}_1,...,\mathbf{t}_N]^T \in \mathbb{R}^{N\times m}$. $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$ (Using Matlab *pinv* function).

3. Experimental Results

In this section, we conduct the performance comparison of the three methods for seven real problems: Digit, DNA, Letter, Sat, Shuttle, Spambase, and Usps. Most of the datasets are taken from the UCI machine learning repository [12]. Usps is taken from database [13]. All the simulations are carried out in MATLAB7.1 environment running in Intel Core(TM) i5-2400, 3.10GHz, 8GBRAM.

LIBSVM is used the libsvm-mat-2.89-3 version [11]. The numbers of attributes, samples for training and testing are shown in Table 1.

In all experiments, the naive QP solver is adopted to solve the QP problem and the Gaussian function is taken as the kernel function $k(x_i, x_j) = \exp(-\|x_i - x_j\|^2 / h)$. where h is the kernel parameter of the Gaussian kernel. The width parameter h is selected to the mean squared norm of the training data, $h = (1/N^2)\sum_{i,j=1}^{N} \|x_i - x_j\|^2$. Setting an appropriate the approximation parameter ε is important in CSELM. The smaller ε will result in more core vectors and the classification speed becomes slower. In our experiment, the activation function in ELM which has better performance is selected from among the *sine*, *sigmoid*, and *RBF* functions. The *sigmoid* function $g(\mathbf{x}) = 1/(1+\exp(-\mathbf{x}))$ was used as an activation function for all models. In the real-world problems, the attributes of their training and testing datasets were scaled to [-1, 1]. The parameters in CSELM and the numbers of hidden nodes in ELM are shown in Table 2. CSELM is the proposed method in this paper. The well-known sequential minimal optimization (SMO) algorithm is used in LIBSVM.

Ten trials were conducted for the three algorithms and the average results are shown in Tables 3 and 4. Table 3 shows the performance comparison of testing accuracy of the three methods in the real-world problems. As observed from the Table 3, general speaking, testing accuracy of CSELM is slightly lower than LIBSVM and ELM methods. This is the reason which we want to improve the classification speed of CSELM, and select the bigger parameter ε . When ε decreases, the testing accuracy becomes higher, and both the number of the core set and the training time increase accordingly. Generally, ε =1e-6 is acceptable in the trade-off of the training speed and the classification accuracy for most cases.

Table 4 shows the performance comparison of average training and testing time of the three methods in the real-world problems. As observed from the Table 4, the learning speed is different; CSELM obtains comparable performance to LIBSVM and ELM methods with much faster learning speed in most datasets. CSELM learns up to 2-10 times faster than ELM in training time. CSELM learns up to 1.5-15 times faster than LIBSVM in training time. Generally, CSELM is faster than LIBSVM and ELM and can reach comparable generalization performance, which could greatly speed up the application running time in many problem domains.

Table 1. Description of datasets

TABLE I. DATASETS	TABLE II. # ATTRIBUTES	TABLE III. # TRAINING	TABLE IV. # TESTING
TABLE V. **DIGIT**	TABLE VI. **64**	TABLE VII. **2810**	TABLE VIII. **2810**
TABLE IX. **DNA**	TABLE X. **180**	TABLE XI. **3457**	TABLE XII. **1729**
TABLE XIII. **LETTER**	TABLE XIV. **16**	TABLE XV. **10000**	TABLE XVI. **10000**
TABLE XVII. **SAT**	TABLE XVIII. **36**	TABLE XIX. **3217**	TABLE XX. **3218**
TABLE XXI. **SHUTTLE**	TABLE XXII. **9**	TABLE XXIII. **29000**	TABLE XXIV. **29000**
TABLE XXV. **SPAMBASE**	TABLE XXVI. **57**	TABLE XXVII. **2300**	TABLE XXVIII. **2301**
TABLE XXIX. **USPS**	TABLE XXX. **256**	TABLE XXXI. **6198**	TABLE XXXII. **3100**

4. Conclusions and Future Work

The GCVM utilizes an approximation algorithm for the center constrain minimum enclosing ball (CC-MEB) problem. We proposed the core set extreme learning machine (CSELM) approach. It consists of two stages. In the first stage, the core set can be obtained efficiently by using the GCVM algorithm. The GCVM algorithm asymptotic time complexity is again linear in N and its space complexity is independent of N, where N is the size of the training patterns. Thus, we can obtain the core set quickly, and the size of the training datasets can be significantly reduced. For the second stage, the extreme learning machine (ELM) can be used to imple-

ment classification. Experiments show that the CSELM has comparable performance with SVM and ELM implementations, but is faster on large datasets.

Further work includes how to select the parameters in algorithms and how to improve the generalization performance of algorithms.

Table 2. Parameters in the experiment

Datasets	# hidden nodes in ELM	ε in CSELM
Digit	400	1e-3
DNA	400	1e-3
Letter	400	1e-4
Sat	400	1e-3
Shuttle	200	1e-3
Spambase	400	1e-3
Usps	400	1e-3

Table 3. Comparison of testing accuracy of the three methods

Datasets	LIBSVM	ELM	CSELM
Digit	99.7865	98.11	93.71
DNA	98.3227	93.06	92.75
Letter	99.1600	87.27	97.08
Sat	98.8813	89.00	91.58
Shuttle	95.76	99.63	98.88
Spambase	87.7879	89.05	87.83
Usps	98.5806	95.26	94.32

Table 4. Comparison of time of the three methods

Datasets	LIBSVM Training testing	ELM Training testing	CSELM Training testing
Digit	0.4212 0.2808	1.3572 0.0780	0.3152 0.0156
DNA	14.2897 5.0700	1.9188 0.0936	0.9156 0.8195
Letter	1.2792 0.8580	4.2900 0.2496	0.4156 0.0312
Sat	0.3588 0.2340	1.6068 0.1092	0.4156 0.0312
Shuttle	35.9114 22.0741	3.5412 0.2808	0.2028 0.0936
Spambase	1.4508 0.9828	1.2012 0.0624	0.0156 0
Usps	31.9490 11.2477	3.0732 0.2184	3.0125 0.0312

Acknowledgments

This research is supported by the National Natural Science Foundation of China (61170040 and 60903089), by the Natural Science Foundation of Hebei Province (F2011201063 and F2012201023), and by the Key Scientific Research Foundation of Education Department of Hebei Province (ZD2010139).

References

[1] G. B. Huang, Q. Y. Zhu, C. K. Siew, "Extreme learning machine: theory and applications," Neurocomputing, 2006, 70：489-501.

[2] R. Zhang, Y. Lan, G. B. Huang, Z. B. Xu, "Universal approximation of extreme learning machine with adaptive growth of hidden node," IEEE Transactions on Neural Networks and Learning Systems, 2012, 23(2):365-371.

[3] C. Panagiotakopoulos, P. Tsampouka, "The Margitron: A Generalized Perceptron with Margin," IEEE Trans. Neural Netw, 2011, 22 (3): 395–407.

[4] L. Yujian, L. Bo, Y. Xinwu, F. Yaozong, L. Houjun, "Multiconlitron: a general piecewise linear classifier," IEEE Trans. Neural Netw. 2011, 22 (2):276–289.

[5] X. Peng, Y. Wang, "Geometric algorithms to large margin classifier based on affine hulls," IEEE Trans. Neural Netw. 2012, 23 (2): 236–246.

[6] J. Platt, B. Schölkopf, C. Burges, and A. Smola, Eds., Fast training of support vector machines using sequential minimal optimization. In Advances in Kernel Methods – Support Vector Learning. Cambridge, MA: MIT Press, 1999, pp. 185–208.

[7] D. Achlioptas, F. McSherry, and B. SchÖlkopf, "Sampling techniques for kernel methods," Advances in neural information processing systems, 2002, 14: 335-342.

[8] I. W. Tsang, J. T. Kwok, and P. M. Cheung, "Core vector machines: fast SVM training on very large data sets," Journal of Machine Learning Research, 2005, 6: 363-392.

[9] I. W. Tsang, J. T. Kwok, and J. M. Zurada, "Generalized core vector machines," IEEE Transactions on Neural Networks, 2006, 17(5): 1126-1140.

[10] W. J. Hu, F. L. Chung, S. T. Wang, "The Maximum Vector Angular Margin Classifier and its fast training on large datasets using a core vector machine," Neural Networks, 2012, 27: 60-73.

[11] C. C. Chang, C. J. Lin, LIBSVM: a library for support vector machines. From: http:// www.csie.ntu.edu.tw/ cjlin/libsvm.

[12] A. Frank, A. Asuncion, UCI machine learning repository, 2010. URL http: //archive.ics.uci.edu/ml.

[18] [13] J. J. Hull, "A database for handwritten text recognition research," IEEE Trans. Pattern Anal. Mach. Intell., 1994, 16 (5), 550-554.