PAPER

Extreme Maximum Margin Clustering

Chen ZHANG[†a)], **ShiXiong XIA**[†], **Bing LIU**[†], ***Nonmembers***, ***and*** **Lei ZHANG**[†], ***Member***

SUMMARY Maximum margin clustering (MMC) is a newly proposed clustering method that extends the large-margin computation of support vector machine (SVM) to unsupervised learning. Traditionally, MMC is formulated as a nonconvex integer programming problem which makes it difficult to solve. Several methods rely on reformulating and relaxing the nonconvex optimization problem as semidefinite programming (SDP) or second-order cone program (SOCP), which are computationally expensive and have difficulty handling large-scale data sets. In linear cases, by making use of the constrained concave-convex procedure (CCCP) and cutting plane algorithm, several MMC methods take linear time to converge to a local optimum, but in nonlinear cases, time complexity is still high. Since extreme learning machine (ELM) has achieved similar generalization performance at much faster learning speed than traditional SVM and LS-SVM, we propose an extreme maximum margin clustering (EMMC) algorithm based on ELM. It can perform well in nonlinear cases. Moreover, the kernel parameters of EMMC need not be tuned by means of random feature mappings. Experimental results on several real-world data sets show that EMMC performs better than traditional MMC methods, especially in handling large-scale data sets.

key words: *maximum margin clustering, unsupervised learning, extreme learning machine (ELM), random feature mapping*

1. Introduction

In machine learning, a recent trend is to incorporate supervised learning with unsupervised learning effectively, i.e. semi-supervised classification, which is actually a supervised method based on clustering or manifold assumptions. More recently, maximum margin clustering (MMC) is a newly proposed clustering method by means of supervised learning method. Different from traditional clustering methods, such as the k-means clustering [1], mixture models [2], and spectral clustering [3], [4], the key idea of MMC is to extend the maximum margin principle of support vector machines (SVM) to the unsupervised learning scenario. Hence the MMC technique often obtains more accurate results than conventional clustering methods.

However, unlike supervised large margin learning methods which can be formulated as a convex optimization problem, MMC is much more computationally difficult. As the labels of samples are unknown, optimization over all possible labeling leads to a hard, non-convex integer optimization problem. Consequently, different optimization techniques have been used to relax the original problem. Xu et al. [5] reformulate it as a semidenite programming (SDP) problem, which could be efficiently solved using standard SDP solvers such as SeDuMi [6] and SDPT3 [7]. Valizadegan and Jin [8] further proposed the generalized MMC (GMMC) algorithm which reduces the number of parameters in the SDP formulation form n^2 to n, where n is the number of samples. This makes MMC more practical in dealing with some data sets. Unfortunately, due to the fact that solving SDP is still computationally expensive, the worst-case time complexity of MMC and GMMC is $O(n^{6.5})$ and $O(n^{4.5})$, respectively. Thus, MMC and GMMC can only handle very small data sets containing several hundreds of samples. Zhang et al. [9] utilized the alternative optimization techniques to solve the MMC problem, in which the MMC result is obtained by solving a series of SVM or Support Vector Regression (SVR) training problems, but it is still hard to handle large-scale data sets. In real-world applications such as image segmentation and text mining, the data set usually contains a large amount of data samples. Therefore, how to make MMC applicable to a large-scale data set is a very challenging and valuable research topic.

Recently, ELM has been attracting considerable interest from more and more researchers [10]–[13]. The idea of ELM is actually the same to that of the random vector functional-link (RVFL) network [14], [15] where the hidden neurons are randomly selected and only the weights of the output layer need to be trained. Hence, ELM can be regarded as the single-hidden-layer RVFL network. Igelnik and Pao [16] proved that the RVFL network is an efficient universal approximator with the rate of approximation error converging to zero of order $O(C/\sqrt{n})$, where n is number of basis functions and with C independent of n. Tyukin et al. [17] proposed that the domain of parameters in RVFL should be bounded, and thus one can assume that basis functions have compact support to mitigate this restriction. On the basis of this conclusion, Huang et al. [18], [19] proved that almost all nonlinear piecewise continuous functions used as feature mapping can make ELM satisfy universal approximation capability, but the convergent rate of approximation error in ELM is hard to estimate. It should also be noticed that approximation error is not at all guaranteed to be close to zero for every randomly chosen set of hidden nodes. Recently, Romero [20] showed that support vector sequential feedforward neural networks have better generalization performance than error minimized extreme learning machines, which build single-hidden-layer feedforward networks sequentially. Fortunately, the relatively fast con-

Manuscript received November 28, 2012.
Manuscript revised March 20, 2013.

[†]The authors are with the School of Computer Science and Technology, China University of Mining and Technology, Xu Zhou, 221116, China.
a) E-mail: zc@cumt.edu.cn

DOI: 10.1587/transinf.E96.D.1745

vergence rate and small approximation error can be guaranteed if the number of hidden nodes is large enough, which is meaningful to large-scale data sets. Like the method used in [21], [22], ELM also applies the zero-order regularization to improve the generalization capacity. Thus, the regularization form of ELM aims to reach not only the smallest training error but also the smallest norm of output weights, which embodies the structural risk minimization principle. In addition, ELM provides a unified solution to different practical applications (e.g., regression, binary, and multiclass classifications), while different variants of LS-SVM and SVM are required for different types of applications, so the application of ELM is much easier.

In this paper, we propose an extreme maximum margin clustering (EMMC) method based on ELM. Firstly, we reformulate the MMC problem based on ELM as a nonconvex optimization problem, and then perform alternating optimization directly on the constructed nonconvex problem instead of relaxing it. Our key modification is to replace SVM or SVR by ELM with the square loss, which can not only speed up the MMC algorithm but also discourage premature convergence. Thus, compared to existing approaches, the proposed EMMC in fact involves only a sequence of ELM training and the resultant implementation is fast and scales well. Experimental evaluations on several real-world data sets show that EMMC performs better than existing MMC methods.

The rest of this paper is organized as follows. In Sect. 2, we briefly introduce some MMC algorithms. In Sect. 3, we briefly introduce the ELM model. The two-class and multiclass EMMC algorithm are presented in detail in Sect. 4. Experimental results on several real-world data sets are provided in Sect. 5, followed by the conclusions in Sect. 6. In order to avoid confusion, we give a list of the main notations used in this paper in Table 1.

Table 1 Notations.

Notation	Explanation
$\mathbb{R}^{\mathrm{d}}$	The input d-dimensional Euclidean space
$\mathbb{R}^{\mathrm{d}}_{+}$	The set of nonnegative vectors in $\mathbb{R}^{\mathrm{d}}$.
$\mathbf{X}$	$\mathbf{X} = [x_1, \ldots, x_n] \in \mathbb{R}^{d \times n}$ is the training data
m	matrix.
y	The number of classes that the samples
$\boldsymbol{Y}$	belong to.
	$\boldsymbol{y} = (y_1, \ldots, y_n) \in \{-1, +1\}^n$ is the 0-1 label vector.
	$\boldsymbol{Y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n) \in \mathbb{R}^{m \times n}$ is the 0-1 label matrix.
$A \succeq 0$	$\boldsymbol{y}_i \in \mathbb{R}^{\mathrm{m}}$ is the lable vector of x_i,
$F(\cdot)$	and all componets of $\boldsymbol{y}_i$ are 0 s except one being 1.
	Matrix A is symmetric and positive semidefinite.
$k(x, y)$	$\boldsymbol{F}(\boldsymbol{x}) = (f_1(x), \ldots, f_m(x))^T$ is the discriminative
$\boldsymbol{K}$	vector function. The index of the class which
$\mathbf{B}$	x belongs to is that of the component with the maximum value.
	Kernel function of variables x and y
$\| \|_K$	Kernel matrix $\boldsymbol{K} = \left\{k\left(x_i, x_j\right)\right\} \in \mathbb{R}^{n \times n}$
$< >_K$	$\mathbf{B} = (\beta_1, \ldots, \beta_n) \in \mathbb{R}^n$. Its columns are the coefficients of the kernel function to represent the discriminative function $F(\cdot)$.
	norm in the Hilbert space H_K
	Inner product in the Hilbert space H_K

2. Maximum Margin Clustering

As stated in the introduction, MMC extends the maximum margin principle of SVM to the unsupervised scenario, which labels the samples by solving the following optimization problem:

$$\min_{y \in \{-1,+1\}^n} \min_{w,b,\xi_i} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i}^{n} \xi_i \tag{1}$$

$$s.t. \quad y_i\left(\boldsymbol{w}^T\varphi(x_i) + b\right) \geq 1 - \xi_i$$

$$\xi_i \geq 0, \qquad i = 1, \ldots, n,$$

where $\boldsymbol{\varphi}(\cdot)$ is a nonlinear mapping induced by the kernel function k, $x_i \in \mathbb{R}^d$, $\varepsilon_i (1 \leq i \leq n)$ is the slack variable for the errors, and $C > 0$ is a tradeoff parameter between the complexity and fitness of the decision function $f(x)$.

It can be observed that the above optimization problem has a trivially "optimal" solution with infinite margin, which is to assign all patterns to the same class. Moreover, another undesirable solution is to separate a very small group of samples or even a single outlier from the rest of data. To avoid these trivial solutions, Xu et al. [5] introduced a class balance constraint on $\boldsymbol{y}$

$$-l \leq \boldsymbol{e}^T\boldsymbol{y} \leq l \tag{2}$$

where $l \geq 0$ is a user-defined constant controlling the class imbalance and $\boldsymbol{e}$ is the all-one vector.

In order to get the solution of problem (1) in reasonable time, Xu et al. [5] proposed to make several relaxations, including relaxing the labeling vector $\boldsymbol{y}$ to take continuous values, relaxing yy^{T} to a positive-semidefinite matrix M with its diagonal elements all set to 1 and setting the bias term b in the decision function to 0, which leads to a SDP problem. In GMMC [8], the number of parameters in the SDP is reduced from n^2 to n and the bias term b may not be 0. Considering the high computational cost of MMC and GMMC, Zhang et al. [9] proposed a simple alternative optimization technique iterative support vector regression (IterSVR), which is more efficient than the SDP-based techniques. However, both GMMC and IterSVR can only tackle two-class problems. Wang et al. [23] propose a cutting plane maximum margin clustering (CPMMC) algorithm. It first decomposes the nonconvex MMC problem into a series of convex subproblems by making use of the constrained concave-convex procedure. Then for each subproblem, it adopts the cutting plane algorithm to solve it. In the linear case, the CPMMC algorithm takes $O(sn)$ time to converge, where s is the sparsity of the data set, i.e., the average number of nonzero features of the data samples. But in the nonlinear case, CPMMC takes $O(T_1T_2n^2)$ time to compute the most violated constraint, where T_1 is the number of CCCP iterations, T_2 is the number of cutting plane iterations. These MMC methods cannot handle large-scale data sets and is often not viable in practice.

3. ELM Model

The output function of ELM for generalized SLFNs in one output node case is

$$f_L(x) = \sum_{i=1}^{L} \beta_i h_i(x) = \boldsymbol{h}(\boldsymbol{x})\boldsymbol{\beta} \tag{3}$$

where $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_L]^T$ is the vector of the output weights between the hidden layer of L nodes and the output node, and $\boldsymbol{h}(\boldsymbol{x}) = [h_1(x), \ldots, h_L(x)]$ is the output (row) vector of the hidden layer with respect to the input x. In fact, $\boldsymbol{h}(\boldsymbol{x})$ actually maps the data from the d-dimensional input space to the L-dimensional hidden-layer feature space H. By imposing a penalty on the norm of $\boldsymbol{\beta}$, ELM minimizes the training error as well as the norm of the output weights [19], [24]

$$\text{Minimize} : \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{T}\|^2 \text{ and } \|\boldsymbol{\beta}\|, \tag{4}$$

where H is the hidden-layer output matrix, denoted by

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{h}(\boldsymbol{x}_1) \\ \boldsymbol{h}(\boldsymbol{x}_2) \\ \vdots \\ \boldsymbol{h}(\boldsymbol{x}_n) \end{bmatrix} = \begin{bmatrix} h_1(x_1) & \ldots & h_L(x_1) \\ h_1(x_2) & \ldots & h_L(x_2) \\ \vdots & \vdots & \vdots \\ h_1(x_n) & \ldots & h_L(x_n) \end{bmatrix} \tag{5}$$

As with SVM for the binary classification, to minimize the norm of the output weights $\|\beta\|$ is actually to maximize the distance of the separating margins of the two different classes in ELM feature space: $2/\|\boldsymbol{\beta}\|$. The norm actually controls the complexity of the function in the ambient space.

If a feature mapping $\boldsymbol{h}(x)$ is unknown to users, the output function of the ELM classifier is

$$\begin{aligned} f(x) &= \boldsymbol{h}(\boldsymbol{x})\boldsymbol{H}^T\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{H}\boldsymbol{H}^T\right)^{-1}\boldsymbol{T} \\ &= [k(x, x_1), \ldots, k(x, x_n)]\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{M}\right)^{-1}\boldsymbol{T} \end{aligned} \tag{6}$$

where $\boldsymbol{M} = \boldsymbol{H}\boldsymbol{H}^T$, $m_(i, j) = k(x_i, x_j)$ and $k(x, y)$ is some kernel function. If a feature mapping $h(x)$ is known, we have

$$\boldsymbol{h}(x) = [G(a_1, b_1, x), \ldots, G(a_L, b_L, x)] \tag{7}$$

where $G(a, b, x)$ is a nonlinear piecewise continuous function satisfying ELM universal approximation capability theorems [25], [26], such as the Sigmoid function $1/(1 + \exp(-(a \cdot x + b)))$ and the Gaussian function $\exp(-b\|x - a\|^2)$, $\{(a_i, b_i)\}_{i=1}^{L}$ are randomly generated according to any continuous probability distribution. The output function of ELM classifier is

$$f(x) = \boldsymbol{h}(x)\boldsymbol{H}^T\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{H}\boldsymbol{H}^T\right)^{-1} \tag{8}$$

or

$$f(x) = \boldsymbol{h}(x)\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{H}^T\boldsymbol{H}\right)^{-1}\boldsymbol{H}^T\boldsymbol{T} \tag{9}$$

where $\boldsymbol{T} = \begin{bmatrix} t_{11} & \ldots & t_{1m} \\ t_{21} & \ldots & t_{2m} \\ \vdots & \vdots & \vdots \\ t_{n1} & \ldots & t_{nm} \end{bmatrix}$ and m is the number of classes.

It can also be noted that the solution of ELM does not include the bias term b in SVM, which will simplify the computation of ELM training. Meanwhile, if $y_i (1 \le i \le n)$ equals to +1 or −1 in ELM, then $V(f, x_i, y_i) = (f_i - y_i)^2 = (1 - f_i y_i)^2$. Thus, to minimize the norm of the output weights $\|\boldsymbol{\beta}\|$ is actually to maximize the distance of the separating margins of the two different classes in ELM feature space. Consequently, we can incorporate ELM into MMC and enhance the performance of MMC by means of ELM.

4. Two-Class and Multiclass EMMC Algorithm

In this section, we will firstly reformulate the MMC problem based on ELM with single output, and then solve the constructed nonconvex MMC problem by means of alternating optimization. Computationally, this allows the nonconvex problem to be formulated as a sequence of ELM training, so the proposed algorithm is fast and effective. Finally, we extend EMMC with single output to the multioutputs scenario.

4.1 EMMC Based on ELM with Single Output

Since ELM can approximate any target continuous functions, the output of the ELM classifier $h(x)\beta$ can be close to the class labels in the corresponding regions as possible. Thus the classification problem for the ELM with a single-output node can be formulated as [13]:

$$\begin{aligned} &\min_{\beta, \xi_i} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \mathrm{C}\frac{1}{2}\sum_{\mathrm{i}=1}^{n}\xi_i^2 \\ &s.t. \quad h(x_i)\boldsymbol{\beta} = t_i - \xi_i, \quad i = 1, \ldots, n, \end{aligned} \tag{10}$$

where $\boldsymbol{h}(\boldsymbol{x}) = [h_1(x), \ldots, h_L(x)]$ is the output (row) vector of the hidden layer with respect to the input x. Thus the corresponding MMC problem is

$$\begin{aligned} &\min_{t, \beta, \xi_i} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \mathrm{C}\frac{1}{2}\sum_{\mathrm{i}=1}^{n}\xi_i^2 \\ &s.t. \quad h(x_i)\boldsymbol{\beta} = t_i - \xi_i, \quad i = 1, \ldots, n, \end{aligned} \tag{11}$$

where $t_i \in \{\pm 1\}$ for two-class clustering with the class balance constraint $-l \le \sum_{i=1}^{n} t_i \le l$ or $t_i \in \{1, \ldots, m\}$ for m-classes clustering with the class balance constraint $-l \le N_p - N_q \le l$, where m is the number of classes, $p, q \in \{1, \ldots, m\}$, N_p and N_q are the number of samples in the pth and qth class, respectively.

A natural way to solve (15) is to use a simple iterative approach based on alternating optimization [9]. This is similar to the Iterative SVR proposed in [27]. First, we fix t and

minimize (15) w.r.t. β, which is just a standard ELM training. Then, we fix β and minimize (15) w.r.t. t. Specifically, we discuss the following problem without the class balance constraint.

$$\begin{aligned} &\min \sum_{i=1}^{n} (h(x_i)\boldsymbol{\beta} - t_i)^2 \qquad (12)\\ &s.t. \quad t_i \in \{\pm 1\} \text{ for two-class clustering}\\ &\text{or} \quad t_i \in \{1, \ldots, m\} \text{ for m-classes clustering.}\\ &i = 1, \ldots, n. \end{aligned}$$

As shown by the following proposition, the problem (16) can be easily solved without the use of any optimization solver.

Proposition 1: For two-class clustering, the optimal strategy to determine t_i's in (16) is to assign all t_i's as -1 for those with $h(x_i)\boldsymbol{\beta} \leq 0$, and assign t_i's as 1 for those with $h(x_i)\boldsymbol{\beta} > 0$; For muticlass clustering, the optimal strategy to determine t_i's in (16) is to assign all t_i's as i for those with $i - 1 < h(x_i)\boldsymbol{\beta} \leq i$, $i \in \{1, \ldots, m\}$, where m is the number of classes.

The proof of Proposition 1 is similar to that of the Iterative SVR proposed in [27], we don't discuss it further.

4.2 EMMC Based on ELM with Multioutputs

If ELM has multioutput nodes, an m-class classifier is corresponding to m output nodes. If the original class label is l, the expected output vector of the m output nodes is $\mathbf{t}_i = [0, \ldots, 0, \overset{l}{1}, 0, \ldots, 0]^{\mathrm{T}}$. That is, the lth element of $\boldsymbol{t}_i = [t_{i1}, \ldots, t_{im}]^T$ is one, while the rest of the elements are set to zero. The classification problem for ELM with multioutput nodes can be formulated as [13]

$$\begin{aligned} &\min_{\beta,\xi_i} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + C\frac{1}{2}\sum_{i=1}^{n} \boldsymbol{\xi}_i^2 \qquad (13)\\ &s.t. \quad h(x_i)\boldsymbol{\beta} = \boldsymbol{t}_i^T - \boldsymbol{\xi}_i^T, \quad i = 1, \ldots, n \end{aligned}$$

where $\boldsymbol{\xi}_i = [\xi_{i,1}, \ldots, \xi_{i,m}]^T$ is the training error vector of the m output nodes with respect to the training sample x_i. The corresponding MMC problem is

$$\begin{aligned} &\min_{t,\beta,\xi_i} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + C\frac{1}{2}\sum_{i}^{n} \boldsymbol{\xi}_i^2 \qquad (14)\\ &s.t. \quad h(x_i)\boldsymbol{\beta} = \boldsymbol{t}_i^T - \boldsymbol{\xi}_i^T, \quad i = 1, \ldots, n \end{aligned}$$

$\boldsymbol{t}_i \in \{[t_{i1}, \ldots, t_{im}]^{\mathrm{T}}\}$ the wth element is one and the rest of the elements are set to zero, $w \in \{1, \ldots, m\}$. where m is the number of classes, $-l \leq N_p - N_q \leq l$, $p, q \in \{1, \ldots, m\}$, N_p and N_q are the number of samples in the pth and qth class, respectively.

Solving Eq. (18) by the alternative optimization method and enforcing the class balance constraint are similar to those of EMMC based on ELM with the single output. The difference is that the output function of ELM with multioutputs is the function vector, i.e., $\boldsymbol{F}(\boldsymbol{x}) = [f_1(x), \ldots, f_m(x)]^T$, Thus we first compute $h(x_i)\boldsymbol{\beta}$, and then assign the labels according to the distance between $h(x_i)\boldsymbol{\beta}$ and $\boldsymbol{t}_i$. Finally, we sort the $\max_i f(x_i)(1 \leq i \leq m)$ and reassign the labels to enforce the class balance constraint.

With Proposition 1, we proceed to enforce the class balance constraint as follows. First, we sort $f(\boldsymbol{x}_i)$ and obtain the number of samples in each class. For two-class clustering, let N_+ and N_- denote the number of positive and negative samples, respectively. If $N_+ - N_- > l$, we change the labels of the first $(N_+ - (n + l)/2)$ smallest positive samples; if $N_+ - N_- < -l$, we change the labels of the last $(N_- - (n + l)/2)$ biggest negative samples. For multiclass clustering, we compute the difference value between N_i and N_j $(i \neq j)$, where N_i denote the number of samples in the ith class, and then find the two class p and q corresponding to the biggest difference value. If $N_p - N_q > l$ and $p > q$, the labels of the first $(N_p - N_q - l)/2$ smallest samples in pth class are changed from p to q. If $N_p - N_q > l$ and $p < q$, the labels of the last $(N_p - N_q - l)/2$ biggest samples in pth class are changed from p to q. If $N_p - N_q < -l$ and $p > q$, the labels of the last $(N_q - N_p - l)/2$ biggest samples in qth class are changed from q to p. If $N_p - N_q < -l$ and $p < q$, the labels of the first $(N_p - N_q - l)/2$ smallest samples in qth class are changed from q to p. This procedure is repeated until the class balance constraint is satisfied.

4.3 EMMC Algorithm

For the sake of clarity, the complete algorithm is summarized in Algorithm 1. In each iteration, the training of ELM takes $O(L^2n + L^3)$ time and the computing and sorting of $h(x_i)\boldsymbol{\beta}$ takes $O(n \log n + nL)$ time, where $L(L \ll n)$ is the number of hidden nodes in ELM. Thus, the iteration process of EMMC takes only $(n \log n + L^2n)$ time. Moreover, the number of iterations in EMMC is usually small (about fifteen in practice).

Algorithm 1: EMMC algorithm

1:Initialize the labels t (e.g., by using a simple clustering algorithm such as k-means).
2:For two-class clustering, fix t, where $t_i \in \{\pm 1\}$ and perform training of ELM with single output. For muticlass clustering, fix t, where $t_i \in \{1, \ldots, m\}$ and perform training of ELM with single output, or fix t, where $t_i \in [t_{i1}, \ldots, t_{im}]^{\mathrm{T}}$ and perform training of ELM with multioutputs.
3:For two-class clustering, assign all t_i's as -1 for those with $h(x_i)\beta \leqslant 0$, and assign t_i's as 1 for those with $h(x_i)\beta > 0$. For muticlass clustering, assign all t_i's as i for those with $i - 1 < h(x_i)\beta \leqslant i$, $i \in \{1, \ldots, m\}$, where m is the number of classes.
4:Check the class balance constraint, if it is violated, sort the $h(x_i)\boldsymbol{\beta}$'s and reassign the labels as described above.
5:Repeat steps 2–4 until convergence.

Hence, EMMC is computationally efficient. It can be noted that EMMC keeps updating all labels. Thus, EMMC can avoid premature convergence by using the square loss [9]. Meanwhile, by enforcing the class balance constraint, it can

handle imbalanced data. For multiclass clustering, ELM with single output or multioutputs can be used in each iteration. For both cases, the hidden-layer matrix $H(9)$ remains the same, and the size of H is only decided by the number of training samples and hidden nodes, which is irrelevant to the number of output nodes (number of classes). Hence, EMMC with single output has comparable performance to that based on multioutputs, which will be validated in Sect. 5.

5. Experiments

In this section, we will validate the performance of the proposed EMMC algorithm on a number of real-world data sets. Specifically, we will analyze how the performance of the proposed algorithm relies on the k-means algorithm. Furthermore, we will study the sensitivity of EMMC to L, both in accuracy and efficiency. All the experiments are performed with MATLAB 7.0.1 environment on a 3.10 GHZ Intel CoreTM i5-2400 with 3-GB RAM.

5.1 Data Sets

We use seven data sets from the UCI machine learning repository (ionosphere, letter, digit and satellite), the LIBSVM data (svmguide1-a), and another benchmark repository (ringnorm, USPS). The same experimental setup was set as in [9]. For the letter and satellite data sets, we use their first two classes only. Several multi-class data sets were created from the digits and letter data. The class balance parameter is always set to $l = 0.03n$ for the balanced data sets, and $l = 0.15n$ for the imbalanced ones. The basic information about these data sets is summarized in Table 2.

5.2 Evaluation Criteria

In the experiments, for the clustering problem, we set the number of clusters equal the true number of classes for all the clustering algorithms. To evaluate their performance, we compare the clusters generated by these algorithms with the true classes by computing the following two performance measures.

Clustering Accuracy (***Acc***) [23]. The clustering accuracy discovers the one-to-one relationship between clusters and classes and measures the extent to which each cluster contained data points from the corresponding class. It sums up the whole matching degree between all pair class clusters.

Table 2 Description of the data sets.

Data	Size(n)	Feature(d)	Class	Balance parameter l
Ionosphere	351	34	2	$0.15n$
LetterA-B	1555	16	2	$0.03n$
SatelliteC1-C2	2236	36	2	$0.15n$
Svmguide1-a	3089	4	2	$0.15n$
Ringnorm	7000	20	2	$0.03n$
Digits0689	713	64	4	$0.03n$
Digits1279	718	64	4	$0.03n$
LetterABCD	3096	16	4	$0.03n$
USPS	9298	256	10	$0.15n$

Clustering accuracy can be computed as

$$Acc = \frac{1}{N} \max_{(C_k, L_m)} \left(\sum_{(C_k, L_m)} T(C_k, L_m) \right) \tag{15}$$

where C_k denotes the kth cluster in the final results, L_m is the true mth class and $T(C_k, L_m)$ is the number of entities which belong to class m and are assigned to cluster k. Accuracy computes the maximum sum of $T(C_k, L_m)$ for all pairs of clusters and classes, and these pairs have no overlaps. It is noted that the greater clustering accuracy means the better clustering performance.

Rand Index (***RI***) [23]. Let $C = \{C_1, C_2, \ldots, C_K\}$ be the set of final clustering results such that C_i represents the ith cluster, and $L = \{L_1, L_2, \ldots, L_K\}$ denotes the set of true data classes such that L_i represents the ith class. The following four variables are defined:

a: the number of data pairs in X that are in the same set in both C and L;

b: the number of data pairs in X that are in different sets in both C and L;

c: the number of data pairs in X that are in the same set in C but different sets in L;

d: the number of data pairs in X that are in different sets in C but the same set in L;

Then, the rand index R that measures the similarity between C and L can be computed as

$$R = \frac{a + b}{a + b + c + d} \tag{16}$$

Intuitively, one can think of $a + b$ as the number of agreements between C and L and $c + d$ as the number of disagreements between C and L. The value of R has a value between 0 and 1, with 0 indicating that C and L do not agree on any pair of data points, and 1 indicating that C and L are exactly the same.

5.3 Experimental Setups and Comparisons

In the experiments, we set the number of clusters equal to the true number of classes for all the clustering algorithms. To evaluate their performance, we compare the clusters generated by these algorithms with the true classes by computing the following two performance measures, i.e., Clustering Accuracy (*Acc*) and Rand Index (*RI*) [24]. We have conducted comprehensive performance evaluations by testing our method and comparing it with the following representative MMC methods on the same data sets:

(1) Maximum Margin Clustering (MMC) [5]. The width of the Gaussian kernel is set by exhaustive search from the grid $\{0.1\sigma_0, 0.2\sigma_0, \ldots, \sigma_0\}$ with σ_0 being the range of distance between any two data points in the data set.

(2) Generalized Maximum Margin Clustering (GMMC) [8]. The experimental setting is the same as in [8].

(3) Cutting Plane Maximum Margin Clustering (CPMMC) [23]. The Gaussian kernel is used in CPMMC and the width of the Gaussian kernel is set in the same way as in MMC.

(4) Iterative Support Vector Regression (IterSVR) [9]. The initialization is based on the k-means with randomly selected initial data centers, and the width of the Gaussian kernel is the same as that of MMC.

(5) Normalized Cut Spectral Clustering (NC) [28]. Using the Gaussian affinity function to construct the graph Laplacian matrix.

(6) Laplacian Regularized Gaussian Mixture Model (LapGMM) [29]. A regularized probabilistic model based on manifold structure for data clustering and the graph structure is incorporated in the maximum likelihood objective function.

It is found that the ELM algorithm achieves good generalization performance as long as L and C are large enough [13]. Thus, we let $C = 500$ in this paper. For both two-class and multiclass EMMC, we use the radial basis function and the results reported in the following are averaged over 20 independent runs. In order to analyze the influence of parameter C and different loss functions, IterSVM with the hinge loss function [9], IterSVR with the squared loss function [9] and the EMMC algorithm were performed on the digit pair of "3" and "9" from the digit data set, respectively. All algorithms used iterative approaches based on alternating optimization. The clustering results are shown in Table 3. As can be seen from Table 3, IterSVR and EMMC perform better than IterSVM, Since IterSVM use the hinge loss function which could cause the premature convergence. By replacing the hinge loss function with the square loss function, IterSVR and EMMC can effectively avoid this phenomenon. With the increase of the parameter C, the performance of EMMC does not vary monotonically. Thus, C is chosen from the range $\{2^{-10}, 2^{-9}, \ldots, 2^{9}, 2^{10}\}$ by cross validation in the next testing.

Firstly, we study the effect of initialization on EMMC with single output. The two initialization schemes are included in the experiment: 1) random; 2) standard k-means clustering (KM). 3) k-means clustering in ELM random feature space. Each scheme is repeated 5 times due to the inherent randomness. For comparison, we use all 45 pairs of the digits 0–9 from the optdigit data set in the UCI machine learning repository. We let L equal the number of training samples. The average performance of these 45 tasks is reported in Table 3. As can be seen from Table 3, the clustering error of the random scheme is close to 50% error, EMMC with random initialization has poor performance with the poor initialization. This is because EMMC relies on local optimization. Thus, it cannot recover from a very poor initialization. Meanwhile, it should be noted that EMMC still improved the clustering accuracy by about 20% because of its robustness. With better initialization provided by k-means clustering algorithm, EMMC is able to obtain a higher clustering accuracy. Similar to kernel k-means clustering algorithms, the performance of the k-means clustering algorithm can be enhanced by the ELM feature mapping. Obviously, EMMC achieves the best clustering accuracy based on the last initialization scheme. Thus, the last initialization scheme was chosen in the next testing.

It should be noted that multiclass EMMC is based on ELM with single output or multiouputs. Hence, we perform the multiclass EMMC algorithm on LetterABCD and USPS data sets in both cases and analyze the influence of different numbers of hidden nodes on the clustering results. As can be seen from Tables 4 and 5, the clustering accuracy of EMMC with single output is slightly lower than that of EMMC with multioutputs, while EMMC with single output performs a little faster than EMMC with multioutputs, which is consistent with the analysis of EMMC in Sect. 4. Thus, for simplicity, we use the EMMC algorithm with single output in both two-class and multiclass clustering, and then compare it with the other MMC algorithms.

We further perform EMMC with different numbers of hidden nodes on several data sets, whose size is bigger than 1000. Figure 1 shows the clustering accuracy of EMMC with various values for L. It can be seen from Fig. 1 that the clustering accuracy of each data set grows quickly as the number of hidden nodes increases, but it begin to grow slowly after the variable L reaches some value. In Fig. 2 the CPU time of EMMC grows nonlinearly with the increase of the variable L. In order to achieve good clustering results at a relatively fast speed, we select the number of hidden nodes

Table 3 Average clustering errors on Digits "3" and "9".

C	0.00001	0.0001	0.001	0.01	0.1	1	100
IterSVM	31.13	21.27	19.7	19.2	19.7	19.46	19.46
IterSVR	14.01	2.98	2.85	3.24	3.76	3.5	3.5
EMMC	13.28	3.25	2.74	3.41	3.58	3.42	3.42

Table 4 Average Performance on the 45 Clustering Tasks Under Different Initialization Schemes.

Clustering Scheme	Clustering error (%)	CPU time (Second)
Random only	48.21	0.001
Random + EMMC	26.37	11.06
Standard KM	3.49	0.025
KM in ELM feature space	2.16	0.28
Standard KM + EMMC	1.84	1.69
KM in ELM feature space+EMMC	1.78	1.95

Table 5 Clustering Results Comparisons between EMMC with single output and multiouputs on LetterABCD data set.

The number of hidden nodes L	**EMMC with single output**		**EMMC with multioutpus**	
	Acc (%)	**Time (s)**	**Acc (%)**	**Time (s)**
100	40.17	3.36	40.86	3.79
150	56.35	4.70	56.25	5.58
200	60.74	6.27	61.44	7.36
300	68.25	13.32	68.73	14.86
500	69.67	19.68	70.85	22.52
1000	71.53	103.76	71.72	109.69

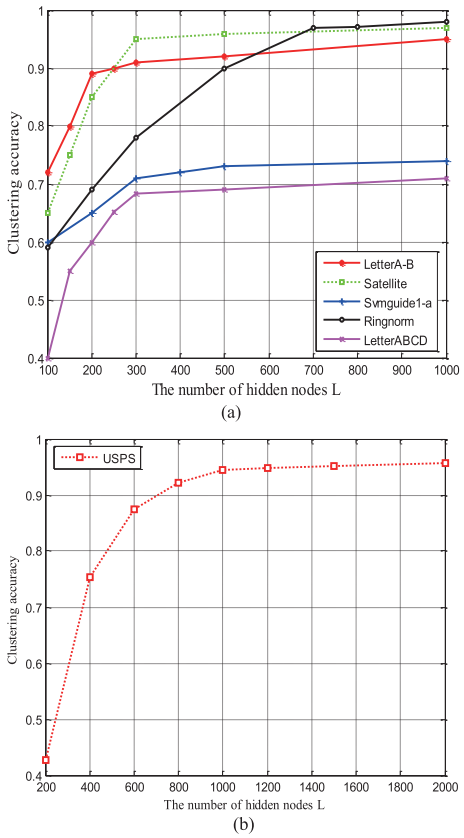

Fig. 1 Clustering accuracy of EMMC with various values for L. (a) several data sets. (b) USPS.

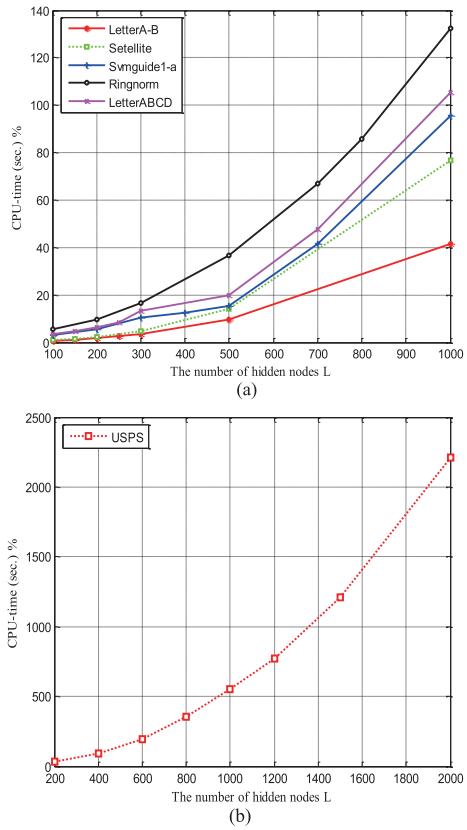

Fig. 2 CPU time (in seconds) of EMMC as a function of the number of hidden nodes. (a) several data sets. (b) USPS.

for each dataset. From Fig. 1 and Fig. 2, it is not difficult to find out that letting L equal 300 is suitable for letterA-B, satelliteC1-C2, svmguide1-a and letterABCD. For ringnorm and USPS, we let L equal 300 and 1000, respectively. For the rest small data sets, since the training of ELM is very fast in EMMC, we let L equal the number of training samples. The clustering results of our algorithm and other competitive methods are shown in Tables 6–8. The symbol '-' means that the corresponding algorithm cannot handle the data set in reasonable time. The symbol '*' means that the corresponding algorithm cannot handle multiclass problems. It can be seen from Tables 6 and 7 that the proposed EMMC algorithm performs better than MMC, GMMC, IterSVR and CPMMC. Among the 7 two-class data sets, EMMC reports 4 best results, which are the largest among all MMC algorithms. The EMMC, IterSVR and CPMMC algorithms are more effective than GMMC and MMC. GMMC is slightly inferior to EMMC. Furthermore, by enforcing the class balance constraint, EMMC can handle imbalanced data sets well, such as Ionosphere, SatelliteC1-C2 and Svmguide1-a. Specifically, for multiclass clustering, the GMMC and IterSVR algorithms cannot handle multiclass problems, while the clustering results of EMMC are also comparable with CPMMC. From Table 8, we can see that EMMC reports 3 best results among the 4 multiclass data sets. Only for the Digits0689 data set,

Table 6 Clustering Results Comparisons between EMMC with single output and multiouputs on USPS data set.

The number of hidden nodes L	**EMMC with single output**		**EMMC with multioutpus**	
	Acc (%)	**Time (s)**	**Acc (%)**	**Time (s)**
200	42.65	30.51	41.31	32.62
400	75.29	87.90	75.54	92.11
600	87.38	190.56	87.74	196.885
800	92.26	342.21	92.45	350.875
1000	94.47	536.61	94.89	547.045
1200	94.75	742.05	95.03	754.705
1500	95.11	1173.06	95.57	1188.76

Table 7 Clustering Accuracy (In Percent) and Rand Index Comparisons for Two-Class Problems (The bold element indicates the best performance).

Data	**MMC**		**GMMC**		**IterSVR**		**CPMMC**		**EMMC**	
	Acc	***RI***	***Acc***	***RI***	***Acc***	***RI***	***Acc***	***RI***	***Acc***	***RI***
Ionosphere	**78.75**	**0.67**	76.50	0.64	70.52	0.55	75.48	0.65	74.73	0.63
Digits1-7	68.75	0.57	97.80	0.96	99.45	0.99	**100**	**1.00**	99.26	0.99
Digits8-9	96.25	0.93	84.00	0.73	96.33	0.93	98.12	0.97	**98.36**	**0.97**
LetterA-B	–	–	–	–	92.80	0.87	95.02	0.92	**95.06**	**0.92**
SatelliteC1-C2	–	–	–	–	96.42	0.93	**98.79**	**0.97**	97.11	0.95
Svmguide1-a	–	–	–	–	83.32	0.72	84.85	0.74	**86.63**	**0.75**
Ringnorm	–	–	–	–	97.48	0.95	98.37	0.97	**98.40**	**0.97**

CPMMC achieves better clustering accuracy than EMMC. Table 9 compare the CPU time of two-class and multiclass EMMC with the others, the number inside the bracket is the average number of iterations in EMMC. As can be seen

Table 8 Clustering Accuracy (In Percent) and Rand Index Comparisons for Multiclass Problems (The bold element indicates the best performance).

Data	MMC		GMMC		IterSVR		CPMMC		EMMC	
	Acc	*RI*	*Acc*	*RI*	*Acc*	*RI*	*Acc*	*RI*	*Acc*	*RI*
Digits0689	94.83	0.94	*	*	*	*	**96.85**	**0.97**	96.25	0.97
Digits1279	91.90	0.91	*	*	*	*	95.27	0.95	**96.37**	**0.97**
LetterABCD	–	–	*	*	*	*	71.76	0.76	**78.76**	**0.79**
USPS	–	–	*	*	*	*	96.13	0.97	**96.20**	**0.97**

Table 9 CPU Time (In Seconds) For Two-class and Multiclass Problems (The bold element indicates the best performance).

Data	MMC	GMMC	IterSVR	CPMMC	EMMC
Ionosphere	–	177.48	0.35	4.13	**0.14(5)**
Digits1-7	–	188.25	0.54	4.89	**0.49(5)**
Digits8-9	–	181.32	0.55	4.15	**0.48(6)**
LetterA-B	–	–	11.21	24.06	**7.55(10.08)**
SatelliteC1-C2	–	–	9.28	32.78	**8.90(6.8)**
Svmguide1-a	–	–	25.38	47.45	**15.34(13.32)**
Ringnorm	–	–	103.87	221.32	**73.02(9.34)**
Digits0689	–	*	*	10.64	**8.48(8.74)**
Digits1279	–	*	*	18.93	**9.65(9.83)**
LetterABCD	–	*	*	129.06	**43.32(14.28)**
USPS	–	*	*	–	**536.61(15.46)**

Table 10 Clustering Accuracy (In Percent) and Rand Index Comparisons for Two-class and Multiclass Problems (The bold element indicates the best performance).

Data	NC		LapGMM		EMMC	
	Acc	*RI*	*Acc*	*RI*	*Acc*	*RI*
Ionosphere	**75.12**	**0.63**	73.56	0.62	74.73	0.63
Digits1–7	97.25	0.95	98.37	0.97	**99.26**	**0.99**
Digits8–9	91.36	0.86	94.62	0.91	**98.36**	**0.97**
LetterA-B	86.24	0.75	90.21	0.90	**95.06**	**0.92**
SatelliteC1-C2	95.73	0.92	**97.54**	**0.95**	97.11	0.95
Svmguide1-a	76.26	0.64	**88.35**	**0.76**	86.63	0.75
Ringnorm	–	–	94.76	0.91	**98.40**	**0.97**
Digits0689	90.26	0.90	89.61	0.87	**96.25**	**0.97**
Digits1279	93.35	0.92	90.36	0.90	**96.37**	**0.97**
LetterABCD	**80.68**	**0.81**	77.36	0.76	78.76	0.79
USPS	–	–	88.62	0.89	**96.20**	**0.97**

from Table 9, the MMC and GMMC algorithms are slowest due to the high computation cost of solving SDPs. For two-class clustering, EMMC is at least 3 times faster than CPMMC; for multiclass clustering, EMMC also performs faster than CPMMC. Specifically, CPMMC cannot handle the larger USPS data set in reasonable time. The proposed EMMC algorithm can converge very fast, we see that there are less than 16 iterations for both two-class and multiclass clustering. Hence EMMC has much better scaling behaviors with the sample size than other MMC algorithms.

Finally, we compare EMMC with traditional clustering algorithms. Experimental results are shown in Table 10. Among the 11 testing data sets, EMMC reports 7 best results, which indicates that margin maximization principle is applicable to clustering. The performance of LapGMM is close to that of NC. Both EMMC and LapGMM can effectively larger data sets, such as the Ringnorm and USPS data sets, while NC cannot be run on these data sets for the sake of insufficient memory of our PC. Thus, EMMC performs better than other MMC and traditional clustering algorithms. It can handle not only two-class but multiclass problems, and has good clustering performance at much faster learning speed.

6. Conclusions

In this paper, we propose an efficient approach for solving MMC via ELM. While traditional MMC algorithms are formulated as SDPs or based on the SVM model, our approach is formulated as a sequence of efficient ELM training. Meanwhile, the symmetric square loss function in ELM discourages premature convergence by penalizing overconfident predictions. It is also noted that our method can handle imbalanced data effectively by enforcing the class balance constraint. Empirically, the clustering performance of EMMC is comparable to that of the other MMC algorithms. Moreover, it is much faster and can handle much larger data sets. In the future, we will study how to extend our clustering method to the semi- supervised learning setting. In addition, in order to enhance the performance of EMMC further, we will combine kernel learning methods with our methods.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant Nos. 50674086 and 61003169.

References

[1] J.A. Hartigan and M.A. Wong, "A k-means clustering algorithm," J. Royal Statistical Society, Series C (Applied Statistics), vol.28, pp.100–108, 1979.
[2] R.A. Redner and H.F. Walker, "Mixture densities, maximum likelihood and the EM algorithm," SIAM Review, vol.26, pp.195–239, 1984.
[3] C. Ding, X. He, H. Zha, M. Gu, and H.D. Simon, A min-max cut algorithm for graph partitioning and data clustering, Proc. 1st Int. Conf. Data Mining 2001, pp.107–114, 2001.
[4] J. Shi and J. Malik, "Normalized cuts and image segmentation," IEEE Trans. Pattern Anal. Mach. Intell., vol.22, no.8, pp.888–905, 2000.
[5] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans, "Maximum margin clustering," Advances in Neural Information Processing Systems 17 (NIPS-17), pp.1537–1544, 2004.
[6] J.F. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," Optimization Methods and Software, vol.11, pp.625–653, 1999.
[7] K.C. Toh, M.J. Todd, and R.H. Tütüncü, "SDPT3—a MATLAB software package for semidefinite programming, version 1.3," Optimization Methods and Software, vol.11, pp.545–581, 1999.
[8] H. Valizadegan and R. Jin, "Generalized maximum margin clustering and unsupervised kernel learning," Advances in Neural Information Processing Systems, vol.19, p.1417, 2007.
[9] K. Zhang, I.W. Tsang, and J.T. Kwok, "Maximum margin clustering made practical," IEEE Trans. Neural Netw., vol.20, no.4, pp.583–596, 2009.
[10] G.B. Huang, K. Mao, C.K. Siew, and D.S. Huang, "Fast modular network implementation for support vector machines," IEEE Trans. Neural Netw., vol.16, no.8, pp.1651–1663, 2005.
[11] C.W. Hsu and C.J. Lin, "A comparison of methods for multiclass support vector machines," IEEE Trans. Neural Netw., vol.13, no.2,

pp.415–425, 2002.
[12] Q. Liu, Q. He, and Z. Shi, "Extreme support vector machine classifier," Lect. Notes Comput. Sci., vol.5012, pp.222–233, 2008.
[13] G.B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," IEEE Trans. Syst. Man Cybern. B: Cybern., vol.42, no.2, pp.513–529, 2012.
[14] Y.H. Pao, G.H. Park, and D.J. Sobajic, "Learning and generalization characteristics of the random vector functional-link net," Neurocomputing, vol.6, pp.163–180, 1994.
[15] L.P. Wang and C.R. Wan, Comments on "The extreme learning machine," IEEE Trans. Neural Netw., vol.19, no.8, pp.1494–1495, 2008.
[16] B. Igelnik and Y.H. Pao, "Stochastic choice of basis functions in adaptive function approximation and the functional-link net," IEEE Trans. Neural Netw., vol.6, no.6, pp.1320–1329, 1995.
[17] I.Y. Tyukin and D.V. Prokhorov, "Feasibility of random basis function approximators for modeling and control," IEEE Conference on Control Applications, (CCA) & Intelligent Control, pp.1391–1396, 2009.
[18] G.B. Huang, D.H. Wang, and Y. Lan, "Extreme learning machines: A survey," Int. J. Machine Learning and Cybernetics, pp.1–16, 2011.
[19] G.B. Huang, Q.Y. Zhu, and C.K. Siew, "Extreme learning machine: theory and applications," Neurocomputing, vol.70, pp.489–501, 2006.
[20] E. Romero and R. Alquéar, "Comparing error minimized extreme learning machines and support vector sequential feedforward neural networks for classification problems," IEEE Trans. Neural Netw., vol.25, no.1, pp.122–129, 2012.
[21] S. Mc Loone and G. Irwin, "Improving neural network training solutions using regularisation," Neurocomputing, vol.37, pp.71–90, 2001.
[22] S. McLoone, M.D. Brown, and G. Irwin, "A. lightbody, A hybrid linear/nonlinear training algorithm for feedforward neural networks," IEEE Trans. Neural Netw., vol.9, no.4, pp.669–684, 1998.
[23] F. Wang, B. Zhao, and C. Zhang, "Linear time maximum margin clustering," IEEE Trans. Neural Netw., vol.21, no.2, pp.319–332, 2010.
[24] G.B. Huang, Q.Y. Zhu, and C.K. Siew, "Extreme learning machine: A new learning scheme of feedforward neural networks," Proc. International Joint Conference on Neural Networks (IJCNN2004), vol.2, pp.985–990, 2004.
[25] G.B. Huang, L. Chen, and C.K. Siew, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," IEEE Trans. Neural Netw., vol.17, no.4, pp.879–892, 2006.
[26] G.B. Huang and L. Chen, "Convex incremental extreme learning machine," Neurocomputing, vol.70, pp.3056–3062, 2007.
[27] L. Xu, D. Wilkinson, F. Southey, and D. Schuurmans, "Discriminative unsupervised learning of structured predictors," Proc. 23rd International Conference on Machine Learning, pp.1057–1064, 2006.
[28] J. Shi and J. Malik, "Normalized cuts and image segmentation," IEEE Trans. Pattern Anal. Mach. Intell., vol.22, no.8, pp.888–905, 2000.
[29] X. He, D. Cai, Y. Shao, H. Bao, and J. Han, "Laplacian regularized Gaussian mixture model for data clustering," IEEE Trans. Knowl. Data Eng., vol.23, no.9, pp.1406–1418, 2011.

Chen Zhang is current a Ph.D candidate at China University of Mining and Technology (CUMT), China. She received her MS degree in Computer Application Technology from CUMT in 2004, and her BS degree in Computer Science from CUMT in 2001. She is currently a lecture at school of Computer Science and Technology, CUMT. Her research interest is computation intelligence and machine learning et al.

ShiXiong Xia is born in 1962, Ph.D. He is a professor at school of Computer Science and Technology in CUMT. He has published more than 60 research papers in journals and international conferences. His research interest is Wireless sensor networks and intelligent information processing et al.

Bing Liu is current a Ph.D candidate at China University of Mining and Technology (CUMT), China. She received her MS degree in Computer Application Technology from CUMT in 2005, and her BS degree in Computer Science from CUMT in 2002. She is currently a lecture at school of Computer Science and Technology, CUMT. His research interest is computation intelligence and machine learning et al.

Lei Zhang received his BS Degree in Aircraft Manufacture from Shen Yang Institute of Aeronautical Engineering, China, in 2000, and his MS Degree in Mechanics Design and Theory from Xi' an University of Architecture and Technology in 2003. He received the PhD Degree in the Department of Computer Application Technology, Nan jing University of Aeronautics and Astronautics in 2006. He is now the associate professor in the school of computer science and technology of China University of mining and technology. His current research interests include mobile computing and data mining.