Journal of Intelligent & Fuzzy Systems 30 (2016) 717–728
DOI:10.3233/IFS-151792
IOS Press

A cross-selection instance algorithm

Junhai Zhai[a,b,*], Ta Li[a] and Xizhao Wang[a]
[a]*Key Lab. of Machine Learning and Computational Intelligence, College of Mathematics and Information Science, Hebei University, Baoding, China*
[b]*College of Mathematics, Physics and Information Engineering, Zhejiang Normal University, Jinhua, China*

Abstract. Motivated by the idea of cross-validation, a novel instance selection algorithm is proposed in this paper. The novelties of the proposed algorithm are that (1) it cross selects the important instances from the original data set with a committee, (2) it can deal with the problem of selecting instance from large data sets. We experimentally compared our algorithm with five state-of-the-art approaches which are CNN, ENN, RNN, MCS, and ICF on 3 artificial data sets and 6 UCI data sets, including 4 large data sets, ranking from 130K to 4898K in size. The experimental results show that the proposed algorithm is very efficient and effective, especially on large data sets.

Keywords: Instances selection, extreme learning machine, K-L divergence, large data sets

1. Introduction

Instance selection also named sample selection is to select a small representative subset from original data set by removing the redundancy instances. In the framework of classification, the purpose of instance selection is to reduce computational complexity of classification algorithm without degenerating its classification accuracy. Since Hart's seminal work (i.e. CNN) [1], many instance selection algorithms have been proposed by different researchers. CNN attempts to find a minimal consistent subset (MCS) of the training set. A consistent subset S of a training set T correctly classifies every instance in T with the same accuracy as T itself [2]. CNN algorithm can ensure that all instances in T are classified correctly by S. However, it does not guarantee that S is a MCS. In addition, CNN is especially sensitive to noise, because noisy instances will usually be misclassified by their neighbors, and thus will be retained [3]. The reduced nearest neighbor (RNN) rule proposed by Gates [4] starts with $S = T$ and removes each instance from S if such a removal does not cause any other instances in T to be misclassified by the instances remaining in S. RNN is computationally more expensive than CNN. The selective nearest neighbor rule (SNN) proposed by Ritter [5] improves CNN and RNN by ensuring that a MCS can be found. SNN is much more complex and the computational time is significantly greater than CNN and RNN. Based on the relative significance of the instances in the training set, Dasarathy proposed an algorithm which can identify MCS [6]. The editing nearest neighbor (ENN) rule proposed by Wilson [3] employs the so called editing rule to remove noisy instances in the training set. The rule is that all instances which are incorrectly classified by their nearest neighbors are assumed to be noisy instances. Based on the concepts of coverage and reach ability, the iterative case filtering (ICF) algorithm was introduced in [2]. The reachable set depends on the distance of an instance from its nearest enemy, and the coverage set of every instance is the list of its associates.

Recently, some new instance selection algorithms were developed by different authors. Nikolaidis et al. proposed a class boundary preserving algorithm [7], which discards center instances while it retains a suitable number of border patterns. Based on the concept

*Corresponding author. Junhai Zhai. Tel.: +86 312 5079351; Fax: +86 312 5079630; E-mail: mczjh@yahoo.com.

of Voronoi cells and enemies, Angiulli proposed a fast nearest neighbor condensation (FCNN) algorithm [8] for large data sets classification. The author claimed that FCNN is order independent, its worst-case time complexity is quadratic, and it is likely to select points very close to the decision boundary. Based on the idea of so-called chain which is a sequence of nearest neighbors from alternating classes, Fayed et al. presented a template reduction algorithm [9], the authors make the point that patterns further down the chain are close to the classification boundary. Li presented a critical pattern selection algorithm by considering local geometrical and statistical information [10]. This algorithm selects both border and edge patterns from the data set.

Most of the instance selection algorithms are tailored for nearest neighbor classifier, so the instances selected with these algorithms are often only suitable for nearest neighbor classifiers. In addition, the computational complexities of these algorithms are also very high, for large data sets some algorithms are impracticable. In order to deal with this problem, motivated by the idea of cross validation, we propose a novel instance selection algorithm, which cross selects the important instances from the original data set with a committee. For large data sets, considering the learning efficiency, we use the single-hidden-layer feed-forward neural networks (SLFNs) trained with extreme leaning machine (ELM) [11] as classifiers in the proposed algorithm. ELM has very fast learning speed and very good generalization ability, and it has been successfully applied in function approximation [12, 13], pattern recognition [14, 15], big data classification [16–18], two comprehensive survey on ELM can be found in [19, 20].

The paper is organized as follows. Some related notions and theoretical background are given in Section 2. The proposed methods are presented in Section 3. Experimental results and analysis are presented in Section 4. Section 5 concludes the paper.

2. Preliminaries

ELM is an efficient and practical learning algorithm used for training the single-hidden-layer feed-forward neural networks (SLFNs) [11]. In ELM the input weights and the hidden layer biases can be chosen randomly, the output weights can be analytically determined with Moore-Penrose generalized inverse [21] of the hidden layeroutput matrix. Unlike other gradient-descent based learning algorithms (such as Back Propagation algorithm [22–24]), ELM does not require iterative techniques to adjust input weights and hidden layer biases during training process. ELM can overcome many drawbacks of the traditional gradient-based learning algorithms such as local minimal, low learning speed by randomly selecting input weights and hidden layer bias [19, 20].

Given a training data set, $D = \{(x_i, y_i)|x_i \in R^d, y_i \in R^k, i = 1, 2, \cdots, n\}$, where x_i is a $d \times 1$ input vector and y_i is a $k \times 1$ target vector, a SLFN with m hidden nodes is formulated as

$$f(x_i) = \sum_{j=1}^{m} \beta_j g(w_j \cdot x_i + b_j),\ i = 1, 2, \cdots, n \quad (1)$$

where $w_j = (w_{j1}, w_{j2}, \cdots, w_{jd})^T$ is the weight vector connecting the jth hidden node with the input nodes. b_j is the threshold of the jth hidden node. w_j and b_j are randomly assigned. $\beta_j = (\beta_{j1}, \beta_{j2}, \cdots, \beta_{jm})^T$ is the weight vector connecting the jth hidden node with the output nodes. The parameters β_j $(j = 1, 2, \cdots, m)$ may be estimated by least-square fitting with the given training data set D, i.e. , satisfying

$$f(x_i) = \sum_{j=1}^{m} \beta_j g(w_j \cdot x_i + b_j) = y_i \quad (2)$$

Equation (2) can be written in a more compact format as

$$H\beta = Y \quad (3)$$

where

$$H = \begin{bmatrix} g(w_1 \cdot x_1 + b_1) & \cdots & g(w_m \cdot x_1 + b_m) \\ \vdots & \ddots & \vdots \\ g(w_1 \cdot x_n + b_1) & \cdots & g(w_m \cdot x_n + b_m) \end{bmatrix} \quad (4)$$

$$\beta = (\beta_1^T, \cdots, \beta_m^T) \quad (5)$$

and

$$Y = (y_1^T, \cdots, y_n^T) \quad (6)$$

H is the hidden layer output matrix of the network, where the jth column of H is the jth hidden nodes output vector with respect to inputs $x_1, x_2, \cdots, x_n$, and the ith row of H is the output vector of the hidden layer with respect to input x_i. If the number of hidden nodes is equal to the number of distinct training samples, the matrix H is square and invertible, and SLFNs can approximate these training samples with zero error. But generally, the number of hidden nodes is much less than the number of training samples. Therefore, H is a non-square matrix and one cannot expect an exact solution of the system (3). Fortunately, it has been proved in [26] that SLFNs with random hidden nodes have the universal approximation capability and the hidden nodes could be randomly generated. The least-square fitting is to solve the following equation.

$$\min_{\beta} = \|H\beta - Y\| \tag{7}$$

The smallest norm least-squares solution of (7) may be easily obtained:

$$\hat{\beta} = H^{\dagger}Y \tag{8}$$

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

The ELM Algorithm [11] is presented as follows.

Algorithm 1 ELM Algorithm

```
Input:
    Training data set $D = \{(x_i, y_i)|x_i \in R^d, y_i \in R^k, i = 1, 2, \cdots, n\}$, an activation function $g$, and the number of hidden nodes $m$.
Output:
    weights matrix $\hat{\beta}$.
1: Randomly assign input weights $w_j$ and $b_j$, $j = 1, 2, \cdots, m$;
2: Calculate the hidden layer output matrix $H$;
3: Calculate output weights matrix $\hat{\beta} = H^{\dagger}Y$.
```

For the trained SLFN, the outputs are transformed into the interval (0, 1) with softmax function [22], the transformed results can be viewed as posterior probability $p(w_k|x_i)$, where w_k is the kth class, and x_i is the ith input vector. The $p(w_k|x_i)$ is calculated as:

$$p(w_k|x_i) = \frac{e^{y_{ij}}}{\sum_{j=1}^{k} e^{y_{ij}}} \tag{9}$$

3. Cross-selection instance algorithm

In this section, we first present the idea of the proposed algorithm, and then present the proposed algorithm.

3.1. The idea of the proposed algorithm

The idea of the proposed algorithm is illustrated as Fig. 1. We firstly partition the data set into n disjoint subsets, for each subset S_i $(i = 1, 2, \cdots, n)$, we use a committee B to select the important instances from S_i. The committee B consists of $n - 1$ SLFNs which are trained on other $n - 1$ subsets. Fig. 2 shows an example of the algorithm for 1 round, the data set used in this example includes 50 instances.

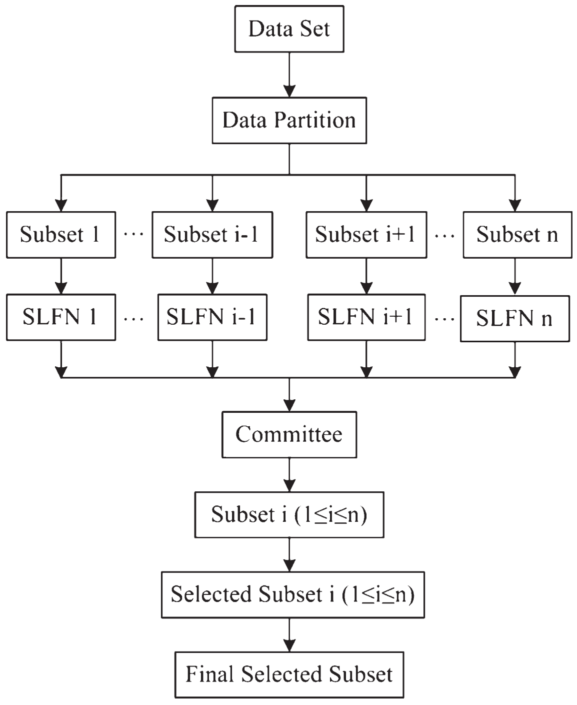


Fig. 1. The idea of the proposed algorithm.

The K-L divergence [26] is employed to measure the significance of the samples, the K-L divergence also called relative entropy is a measure of distance between two distributions [26]. Let x be a discrete random variable, $p(x)$ and $q(x)$ are its two probability mass functions, the definition of K-L divergence is given as follows [26].

$$D(p|q) = \sum_{x \in V} p(x) \log \frac{p(x)}{q(x)} \tag{10}$$

In the above definition, we let $0 \log_2 \frac{0}{0} = 0$, $0 \log_2 \frac{0}{q} = 0$ and $p \log_2 \frac{p}{0} = \infty$.

Let $SLFN_1, SLFN_2, \cdots, SLFN_n$ are the members of the committee B. The informative instances are selected with the following criteria.

$$x^* = \arg\max_{x}\{avg(D(P_{SLFN_i}|P_B))\} \tag{11}$$

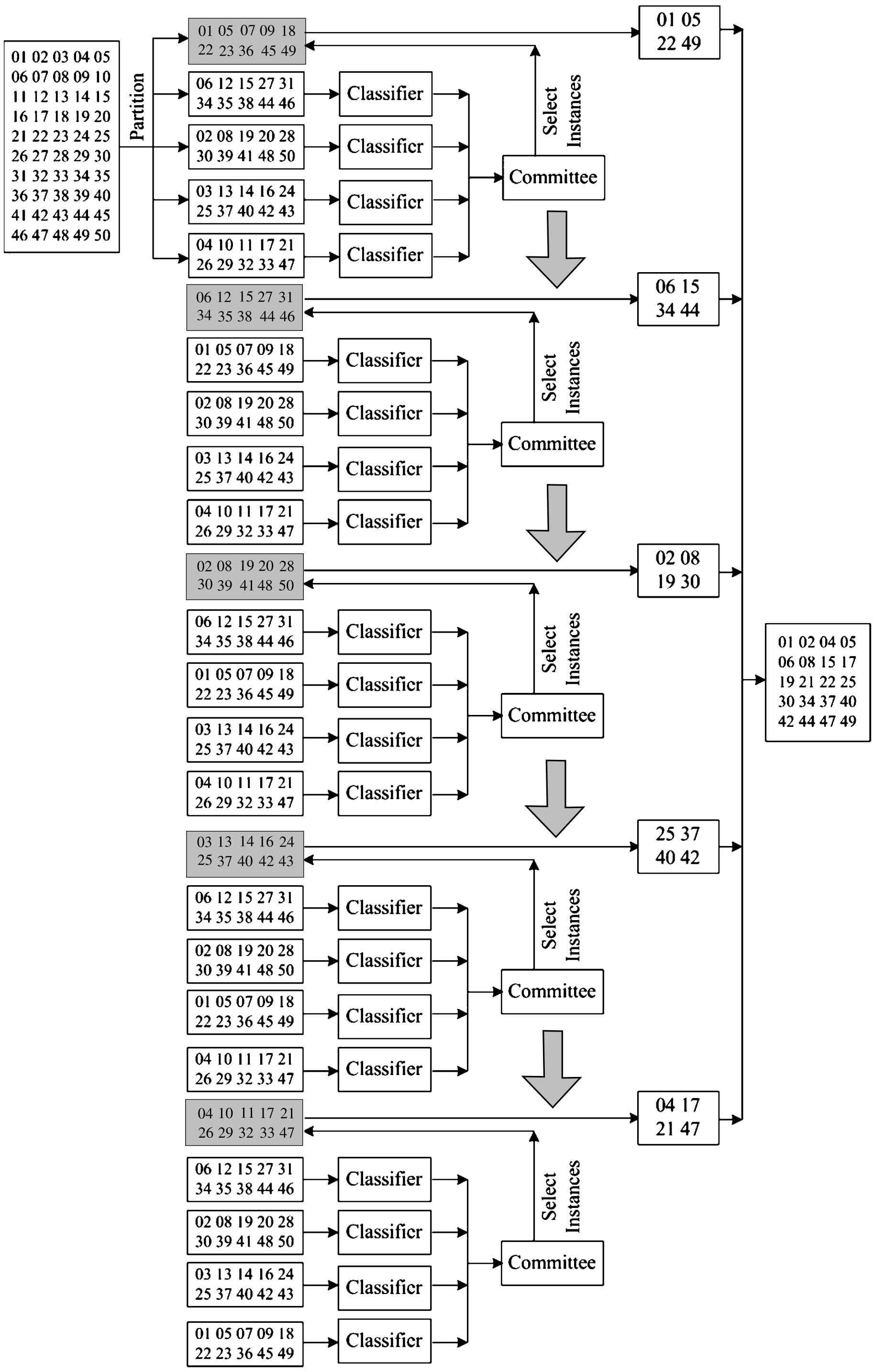


Fig. 2. An example of the algorithm for 1 round.

where

$$avg(D(P_{SLFN_i}|P_B)) = \frac{1}{n}\sum_{i=1}^{n} D(P_{SLFN_i}|P_B) \quad (12)$$

$$D(P_{SLFN_i}|P_B) = \sum_{k=1}^{k} P_{SLFN_i}(w_k|x)\log_2 \frac{P_{SLFN_i}(w_k|x)}{P_B(w_k|x)} \quad (13)$$

$$P_B(w_k|x) = \frac{1}{n}\sum_{i=1}^{n} P_{SLFN_i}(w_k|x) \quad (14)$$

3.2. *The cross-selection instance algorithm*

The proposed algorithm named ELM-KL is described as follows.

Algorithm 2 ELM-KL Algorithm

```
Input:
    $D$, original data set; $p$, the number of partition; $l$, the number of selected instances in a loop; $c_0$, threshold.
Output:
    $S$: selected subset.
 1: Partition $D$ into training set $D_1$ and validation set $D_2$;
 2: Partition training set $D_1$ into $p$ disjoint subsets $S_1, S_2, \cdots, S_p$;
 3: Let $count = 0$, $S = \varnothing$;
 4: For each $S_i$, train $p-1$ $SLFN_j (j \neq i)$ with other $p-1$ subsets $S_j (j \neq i)$, $p-1$ $SLFN_j$ constitute a committee $B$;
 5: For each $x \in S_i$ and each class $\omega_k$, Calculate $P_B(w_k|x)$ with (14);
 6: For each $SLFN_i \in B$, Calculate $D(P_{SLFN_i}|P_B)$ with (13);
 7: Calculate $avg(D(P_{SLFN_i}|P_B))$ with (12);
 8: Select $l$ instances with (11), constitute a subset $S_t$;
 9: Train a SLFN with ELM on subset $S \cup S_t$;
10: Calculate the validation accuracy of SLFN trained with ELM on subset $S \cup S_t$;
11: If($V_a(S \cup S_t) > V_a(S)$), then $S = S \cup S_t$ and $count = 0$; else $count = count + 1$;
12: Calculate $D_1 = D_1 - S_t$;
13: If($count \geq c_0$), then Output $S$; else Goto 2;
```

In the ELM-KL algorithm, $V_a(S)$ denotes the validation accuracy of a classifier which is trained on data set S. We also developed another instance selection algorithm named ELM-KL-ALL, which is similar to ELM-KL, the only difference is that ELM-KL-ALL does not introduce validation set, the trained ELM classifier are validated on training set itself.

4. Experiments and the analysis of the experimental results

4.1. *Experiments settings and the experimental results*

The effectiveness of our proposed method is verified through numerical experiments in the environment of Matlab 7.0 on a Pentium 4 PC. In our experiments we totally select 3 artificial data sets and 6 UCI data sets [27]. The 3 artificial data sets are mainly used to verify the feasibility of the proposed algorithm. The 6 UCI data sets are used to verify the effectiveness and efficiency of the proposed algorithm, we select data sets letter and shuttle because that they contain many categories, the number of classes of the two data sets are 27 and 7 respectively, we select data sets MiniBooNE, skin, artificial_2 and cod_rna because that they are large data sets. The basic information of the 6 UCI data sets is listed in Table 1.

The first artificial data set is two-dimensional concentric data with two classes which are uniform concentric circular distributions [28]. The points of the class ω_1 are uniformly distributed into a circle of radius 0.3 centered on (0.5, 0.5). The points of the class ω_2 are uniformly distributed into a ring centered on (0.5, 0.5) with internal and external radii equal to 0.3 and 0.5, respectively.

The second artificial data set is two-dimensional cloud data with two equal priori probable classes [28]. The class ω_1 is the sum of three different Gaussian distributions:

$$p(x|\omega_1) = \frac{1}{2}\left(\frac{p_1(x)}{2} + \frac{p_2(x)}{2} + p_3(x)\right) \quad (15)$$

Table 1
The basic information of the 6 UCI data sets

Data sets	#Instances	#Attributes	#Classes
letter	18570	16	26
shuttle	58000	9	7
MiniBooNE	130064	50	2
skin	245057	3	2
artificial_2	250000	10	2
cod_rna	488565	8	2

Where, $x = (x_1, x_2)$, and

$$p_i(x) = \frac{1}{2\pi\sigma_{ix_1}\sigma_{ix_2}} \times \exp\left(-\frac{(x_1-\mu_{ix_1})^2}{2\sigma_{ix_1}^2} - \frac{(x_2-\mu_{ix_2})^2}{2\sigma_{ix_2}^2}\right) \tag{16}$$

where μ_{ix_1} and μ_{ix_2} are the means of the components x_1 and x_2 of the ith distribution, σ_{ix_1} and σ_{ix_2} are the corresponding standard deviations.
The class ω_2 is a single Gaussian distribution:

$$p(x|\omega_2) = \frac{1}{2\pi}\exp\left(-\frac{x_1^2+x_2^2}{2}\right) \tag{17}$$

The third artificial data set is a three-dimensional Gaussian data denoted Gaussian with four classes ω_i $(i = 1, 2, 3, 4)$, the distribution of ω_i is $p(x|\omega_i) \sim N(\mu_i, \Sigma_i)$

Where,

$$\mu_1 = (0, 0, 0),\ \Sigma_1 = \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}$$

$$\mu_2 = (0, 1, 0),\ \Sigma_2 = \begin{bmatrix}1&0&1\\0&2&2\\1&2&5\end{bmatrix}$$

$$\mu_3 = (-1, 0, 1),\ \Sigma_3 = \begin{bmatrix}2&0&0\\0&6&0\\0&0&1\end{bmatrix}$$

$$\mu_4 = (0, 0.5, 1),\ \Sigma_4 = \begin{bmatrix}2&0&0\\0&2&0\\0&0&3\end{bmatrix}$$

The basic information of the 3 artificial data sets is listed in Table 2. In our experiments, the values of the attributes of all the data sets are normalized into [–1, 1]. Two thirds of the instances are used as training set, and one third of the instances are used as testing set.

The performances of the proposed algorithms ELM-KL and ELM-KL-ALL are compared with original ELM (ORI in short) and ELM-NN on 4 aspects: the number of selected instances, the optimal number of hidden nodes of SLFNs, the testing accuracy, and the training time. ELM-NN is another instance selection algorithm proposed in our previous work [29]. The experimental results on 3 artificial data sets are listed in Table 3, the experimental results on 6 UCI data sets are listed in Table 4.

From the experimental results listed in Tables 3 and 4, we can find that although the testing accuracies of the proposed algorithm trained on the selected subset are lower than the ones trained on the whole data set on all ten data sets, the difference of the experimental results are slight, and the requirements of memory space and the training time are much less than the ones needed on the whole data set. In the framework of the competence preservation, the proposed algorithm is very effective and efficient. The numbers of optimal hidden nodes in Tables 3 and 4 are determined by using the method proposed in our previous work [30]. The curves described the relationship between the testing accuracy and the number of hidden nodes on the subsets selected from the 3 artificial data sets, and the 6 UCI data sets with ELM-EN, ELM-KL, ELM-KL-ALL and original ELM (ORI in short) are shown in Figs.3 and 4 respectively.

We also compare our algorithm with five state-of-the-art approaches CNN, ENN, RNN, MCS, and ICF. The comparisons of performances on 3 artificial data sets are listed in Tables 5 to 7. The comparisons of performance on the 6 UCI data sets are listed in Tables 8 to 13. where "-" means that the results cannot be obtained. Compared with CNN, ENN, and RNN, our algorithm removes much more instances while obtaining the similar accuracies. Although ICF and MCS have lower selected ratio than our algorithm, they run much slowly than ours. Our algorithm has a compromise between time and selected ratio. What's more, four classical algorithms ENN, RNN, ICF, MCS do not work out results on the two largest data set selected in our experiments, i.e. artificial_2 and cod_rna. Two classical algorithms MCS and ICF do not work out results on data sets Shuttle and MiniBooNE. RNN does not work out results on artificial data set gaussian. Our proposed algorithms can work well on large data sets.

Table 2
The basic information of the 3 artificial data sets

Data sets	#Instances	#Attributes	#Classes
concentric	10000	2	2
cloud	10000	2	2
gaussian	40000	3	4

4.2. *The analysis of the experimental results*

In this section, we present a theoretical analysis of the computational time complexity of the proposed algorithm ELM-KL to specify the possible reasons of

Table 3
The experimental results on the 3 artificial data sets

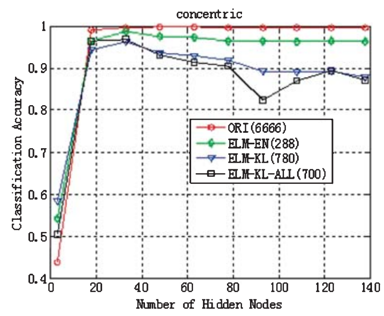
Data sets	Algorithms	#Selected instances	#Optimal hidden nodes	Testing accuracy	Training time
concentric	ORI	6666	42	0.9978	0.0694
	ELM-EN	288	33	0.9860	0.0087
	ELM-KL	780	27	0.9792	0.0087
	ELM-KL-ALL	700	39	0.9692	0.0143
cloud	ORI	6666	90	0.9008	0.4999
	ELM-EN	288	85	0.8596	0.0182
	ELM-KL	1680	65	0.8928	0.0409
	ELM-KL-ALL	1080	95	0.8882	0.0504
gaussian	ORI	26666	155	0.5727	4.5966
	ELM-EN	1056	25	0.5245	0.0221
	ELM-KL	1560	85	0.5636	0.0942
	ELM-KL-ALL	2220	110	0.5684	0.1586

Table 4
The experimental results on the 6 UCI data sets

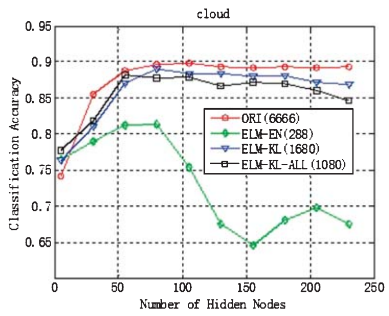
Data sets	Algorithms	#Selected instances	#Optimal hidden nodes	Testing accuracy	Training time
letter	ORI	12380	1500	0.9501	61.8324
	ELM-EN	5820	1800	0.9299	91.0960
	ELM-KL	7320	2100	0.9492	131.9845
	ELM-KL-ALL	8040	2100	0.9515	133.6210
shuttle	ORI	38666	390	0.9973	10.8409
	ELM-EN	1200	120	0.9944	0.1541
	ELM-KL	1710	180	0.9894	0.2452
	ELM-KL-ALL	2940	180	0.9941	0.3602
MiniBooNE	ORI	86709	252	0.9170	19.2086
	ELM-EN	1200	180	0.9108	0.4199
	ELM-KL	1020	276	0.8862	0.6431
	ELM-KL-ALL	6132	540	0.9207	4.5385
skin	ORI	163371	90	0.9927	16.4253
	ELM-EN	7000	180	0.9925	1.7672
	ELM-KL	15000	180	0.9808	3.4153
	ELM-KL-ALL	27000	90	0.9889	2.5708
artificial_2	ORI	166666	22	0.7191	1.3610
	ELM-EN	1500	48	0.6983	0.1734
	ELM-KL	480	30	0.7178	0.1037
	ELM-KL-ALL	2916	32	0.7187	0.1240
cod_rna	ORI	325710	68	0.9597	16.1655
	ELM-EN	2000	68	0.9556	0.4141
	ELM-KL	10000	132	0.9583	1.4635
	ELM-KL-ALL	19764	96	0.9609	1.8600

Table 5
The experimental results on the data set concentric

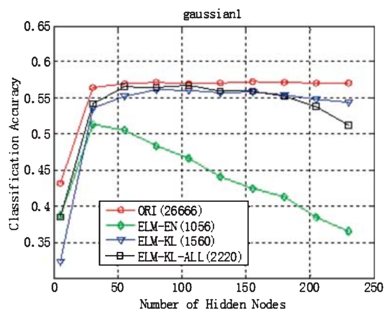
Algorithms	CPU time	Testing accuracy	#Selected instance	Selected ratio
CNN	5.8510	0.9820	240	0.0360
ENN	8.2481	0.9868	6601	0.9902
RNN	336.3406	0.9820	227	0.0341
MCS	1511.7455	0.9826	271	0.0407
ICF	1398.1360	0.9862	515	0.0773
ELM-KL	10.0628	0.9792	780	0.1170
ELM-KL-ALL	6.6688	0.9692	700	0.1050



(a1) The curves of testing accuracy on dataset concentric

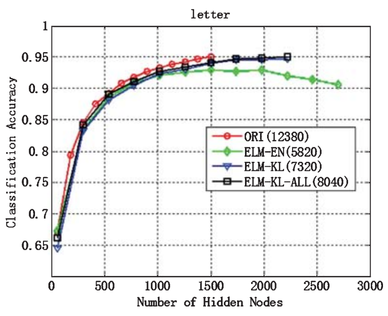


(a2) The curves of testing accuracy on dataset cloud

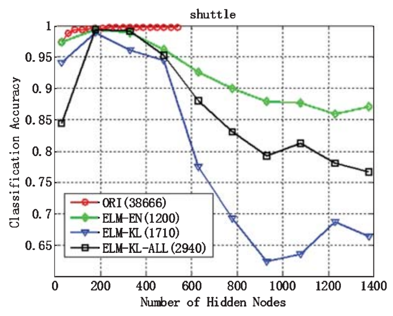


(a3) The curves of testing accuracy on dataset gaussian

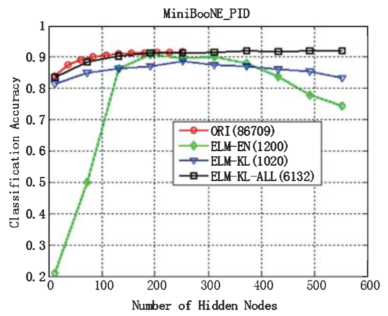
Fig. 3. The curves of testing accuracy on the 3 artificial data sets.



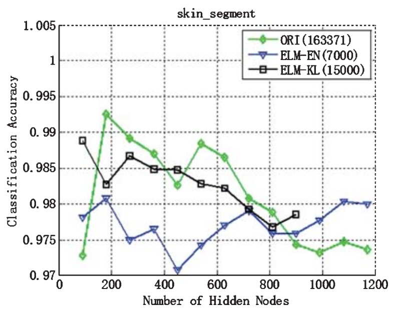
(b1) The curves of testing accuracy on dataset letter



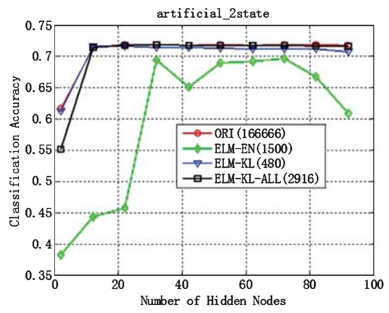
(b2) The curves of testing accuracy on dataset shuttle



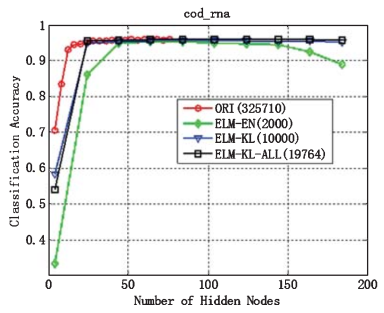
(b3) The curves of testing accuracy on dataset MiniBooNE



(b4) The curves of testing accuracy on dataset skin



(b5) The curves of testing accuracy on dataset artificial_2



(b6) The curves of testing accuracy on dataset cod_rna

Fig. 4. The curves of testing accuracy on the 6 UCI data sets.

the results of the experiment. The ELM-KL (algorithm 2) consists of 13 steps, it is obviously, the computational complexity of the step 1, 2 and 3 are $O(n)$, $O(n)$ and $O(1)$ respectively. The step 4 of algorithm 2 is actually to train p SLFN with ELM algorithm, it is well known that the main computational cost of ELM comes from the calculation of the Moore-Penrose generalized inverse of hidden layer output matrix H. Huang et al. [11] pointed out that, when the n training samples are distinct, the hidden-layer output matrix H is column full rank with probability one, so the ELM can be solved as a full-rank least-square problem [31]. For such a problem, some methods, such as orthogonal project, Householder triangularization, and Gram-Schmidt orthogonalization, may be used to solve it, with computational time complexity $O(m^2n)$ [21]. Hence, the computational time complexity of ELM algorithm is $O(m^2n)$, where m is the number

Table 6
The experimental results on the data set cloud

Algorithms	CPU time	Testing accuracy	#Selected instance	Selected ratio
CNN	7.6715	0.8350	1725	0.2588
ENN	7.4860	0.8872	5671	0.8507
RNN	3180.4889	0.8350	1721	0.2582
MCS	1196.2013	0.8734	1029	0.1544
ICF	910.5015	0.8824	775	0.1163
ELM-KL	17.5268	0.8928	1680	0.2520
ELM-KL-ALL	15.5289	0.8882	1080	0.1620

Table 7
The experimental results on the data set gaussian

Algorithms	CPU time	Testing accuracy	#Selected instance	Selected ratio
CNN	118.5483	0.4494	18449	0.6919
ENN	180.2716	0.5193	12195	0.4573
RNN	–	–	–	–
MCS	41967.1556	0.5071	6844	0.2567
ICF	21739.3537	0.5134	6639	0.2490
ELM-KL	88.8690	0.5636	1560	0.0585
ELM-KL-ALL	100.2332	0.5684	2220	0.0833

Table 8
The experimental results on the data set letter

Algorithms	CPU time	Testing accuracy	#Selected instance	Selected ratio
CNN	39.9782	0.9102	2262	0.1827
ENN	114.1854	0.9367	11768	0.9506
RNN	16002.1794	0.9102	2260	0.1826
MCS	7037.5992	0.9092	2026	0.1637
ICF	9009.4445	0.9283	5083	0.4106
ELM-KL	5450.5569	0.9492	7320	0.5913
ELM-KL-ALL	6978.5808	0.9515	8040	0.6494

Table 9
The experimental results on the data set shuttle

Algorithms	CPU time	Testing accuracy	#Selected instance	Selected ratio
CNN	37.0411	0.9989	174	0.0045
ENN	737.4826	0.9988	38630	0.9991
RNN	1567.1027	0.9989	160	0.0041
MCS	–	–	–	–
ICF	–	–	–	–
ELM-KL	258.7930	0.9894	1710	0.0442
ELM-KL-ALL	564.2342	0.9941	2940	0.0760

Table 10
The experimental results on the data set MiniBooNE

Algorithms	CPU time	Testing accuracy	#Selected instance	Selected ratio
CNN	13084.5546	0.8296	24857	0.2867
ENN	16675.7168	0.8759	74305	0.8569
RNN	–	–	–	–
MCS	–	–	–	–
ICF	–	–	–	–
ELM-KL	1369.0141	0.8862	1020	0.0118
ELM-KL-ALL	9923.2437	0.9207	6132	0.0707

Table 11
The experimental results on the data set skin

Algorithms	CPU time	Testing accuracy	#Selected instance	Selected ratio
CNN	166.7861	0.9995	377	0.0023
ENN	8379.5660	0.9997	163287	0.9995
RNN	15106.0892	0.9995	336	0.0021
MCS	–	–	–	–
ICF	–	–	–	–
ELM-KL	993.5088	0.9808	15000	0.0918
ELM-KL-ALL	11422.5497	0.9889	27000	0.1653

Table 12
The experimental results on the data set artificial_2

Algorithms	CPU time	Testing accuracy	#Selected instance	Selected ratio
CNN	19924.9839	0.5872	96213	0.5773
ENN	–	–	–	–
RNN	–	–	–	–
MCS	–	–	–	–
ICF	–	–	–	–
ELM-KL	289.8407	0.7178	480	0.0029
ELM-KL-ALL	665.9558	0.7187	2916	0.0175

Table 13
The experimental results on the data set cod_rna

Algorithms	CPU time	Testing accuracy	#Selected instance	Selected ratio
CNN	24811.9301	0.9485	42769	0.1313
ENN	–	–	–	–
RNN	–	–	–	–
MCS	–	–	–	–
ICF	–	–	–	–
ELM-KL	2392.7120	0.9583	10000	0.0307
ELM-KL-ALL	25785.4931	0.9609	19764	0.0607

Table 14
The computational time complexity of CNN, ENN, RNN, MCS, ICF and ELM-KL

Algorithms	CNN	ENN	RNN	MCS	ICF	ELM-KL
Computational time complexities	$O(n^2)$	$O(qn^2)$	$O(n^3)$	$O(n^3)$	$O(qn^2)$	$O(pn)$

of attributes. When $m \ll n$, the computational time complexity of ELM can be thought to approach $O(n)$ [31]. Accordingly, the computational complexity of the step 4 is $O(pn)$. It is easy to find that the computational time complexity of step 5, 6, 7 and 8 are $O(n)$, $O(kn)$, $O(p)$ and $O(lp)$, where k is the number of classes. Obviously, the computational time complexity of step 9, 10, 11, 12 and 13 are $O(lp)$, $O(1)$, $O(1)$, $O(1)$ and $O(1)$ respectively. Hence, the computational time complexity of the proposed algorithm is $O(n) + O(n) + O(1) + O(\frac{n}{p}) + O(n) + O(kn) + O(p) + O(lp) + O(lp) + O(1) + O(1) + O(1) + O(1)$. Generally, the number of classes k and the number of selected instances in a loop l are far less than the number of instances of training set n, so the computational time complexity of the proposed algorithm is $O(pn)$.

The computational time complexity of CNN, ENN, RNN, MCS, and ICF are $O(n^2)$, $O(qn^2)$, $O(n^3)$, $O(n^3)$ and $O(qn^2)$ [32], where q is the nearest numbers. For convenient comparison, the computational time complexities of the 6 algorithms are summarized in Table 14.

Based on the above analysis, it can be seen from Table 14 that the computational time complexity of ELM-KL is the minimum among the 6 algorithms. The consequence of theoretical analysis justify the experimental results, for example, it can be seen from Tables 12 and 13 that four classical algorithms ENN, RNN, ICF, MCS do not work out results on the two largest data set selected in our experiments, i.e. artificial_2 and cod_rna. The possible reasons is that the computational time complexity of the four classical algorithms are to high to work out results on large data sets.

5. Conclusions

Motivated by the idea of cross-validation, in this paper, we proposed two instance selection algorithms which are practicable in large data sets, while some classical algorithms (e.g. ENN, RNN, ICF, MCS) are impracticable. Further more, the proposed algorithms have two advantages: fast leaning speed and low selected ratio. The low selected ratio is achieved by discarding the new selected instances which cannot increase the testing accuracy in each loop. The fast leaning speed is due to the choice of extreme learning machine. The experimental results have verified that the proposed algorithms are much more feasible and effective than five state-of-the-art approaches CNN, ENN, RNN, MCS, and ICF. There are two problems related to our work are worth further investigating, the one problem is how to deploy the proposed algorithms to cloud computing environment, such as, Hadoop MapReduce environment? The other problem is how to scale the proposed algorithms to imbalanced data sets? especially imbalanced big data.

Acknowledgments

This research is supported by the national natural science foundation of China (61170040, 71371063), by the natural science foundation of Hebei Province (F2013201110, F2013201220), by the Key Scientific Research Foundation of Education Department of Hebei Province (ZD20131028) and by the Opening Fund of Zhejiang Provincial Top Key Discipline of Computer Science and Technology at Zhejiang Normal University, China.

References

[1] P. Hart, The condensed nearest neighbor rule, *IEEE Transaction on Information Theory* **14**(5) (1967), 515–516.
[2] B. Brighton and C. Mellish, Advances in instance selection for instance-based learning algorithms, *Data Mining and Knowledge Discovery* **6**(2) (2002), 153–172.
[3] D.R. Wilson and T.R. Martinez, Reduction techniques for instance-based learning algorithms, *Machine Learning* **38**(3) (2000), 257–286.
[4] G.W. Gates, The reduced nearest neighbor rule, *IEEE Transactions on Information Theory* **18**(3) (1972), 431–433.
[5] G.L. Ritter, H.B. Woodruff and S.R. Lowry, An algorithm for a selective nearest neighbor decision rule, *IEEE Transactions on Information Theory* **21**(6) (1975), 665–669.
[6] B.V. Dasarathy, Minimal consistent set (MCS) identification for optimal nearest neighbor decision systems design, *IEEE Transactions on Systems, Man, And Cybernetics* **24**(1) (1994), 511–517.
[7] K. Nikolaidis, J.Y. Goulermas and Q.H. Wu, A class boundary preserving algorithm for data condensation, *Pattern Recognition* **44**(3) (2011), 704–715.
[8] F. Angiulli, Fast nearest neighbor condensation for large data sets classification, *IEEE Transactions on Knowledge and Data Engineering* **19**(11) (2007), 1450–1464.
[9] H.A. Fayed and A.F. Atiya, A novel template reduction approach for the k-nearest neighbor method, *IEEE Transactions on Neural Networks* **20**(5) (2009), 890–896.
[10] Y.H. Li and L. Maguire, Selecting critical patterns based on local geometrical and statistical information, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **33**(6) (2011), 1189–1201.
[11] G.B. Huang, Q.Y. Zhu and C.K. Siew, Extreme learning machine: Theory and applications, *Neurocomputing* **70**(1-3) (2006), 489–501.
[12] G.B. Huang, H.M. Zhou, X.J. Ding and R. Zhang, Extreme learning machine for regression and multiclass classification, *IEEE Transactions on Systems, Man, and Cybernetics, Part B* **42**(2) (2012), 513–529.
[13] Y. Lan, Y.C. Soh and G.B. Huang, Constructive hidden nodes selection of extreme learning machine for regression, *Neurocomputing* **73** (2010), 3191–3199.
[14] B.P. Chacko, V.R.V. Krishnan, G. Raju and P.B. Anto, Handwritten character recognition using wavelet energy and extreme learning machine, *International Journal of Machine Learning and Cybernetics* **3**(2) (2012), 149–161.
[15] J. Wu, S.T. Wang and F.L. Chung, Positive and negative fuzzy rule system, extreme learning machine and image classification, *International Journal of Machine Learning and Cybernetics* **2**(4) (2011), 261–271.
[16] B. Wang, S. Huang, J. Qiu, Y. Liu and G. Wang, Parallel online sequential extreme learning machine based on MapReduce, *Neurocomputing* **149**(Part A) (2015), 224–232.
[17] R. Wang, Y.L. He, C.Y. Chow, F.F. Ou and J. Zhang, Learning ELM-Tree from big data based on uncertainty reduction, *Fuzzy Sets and Systems* **258** (2015), 79–100.
[18] J. Chen, H. Chen, X. Wan and G. Zheng, MR-ELM: A MapReduce-based framework for large-scale ELM training in big data era, *Neural Computing and Applications* (2014). DOI:10.1007/s00521-014-1559-3
[19] G.B. Huang, D.H. Wang and Y. Lan, Extreme learning machines: A survey, *International Journal of Machine Learning and Cybernetics* **2**(2) (2011), 107–122.
[20] G. Huang, G.B. Huang, S. Song and K. You, Trends in extreme learning machines: A review, *Neural Networks* **61** (2015), 32–48.
[21] G.H. Golub and C.F.V. Loan, Matrix Computations, 3rd ed. Baltimore, MD: The Johns Hopkins University Press, 1996.
[22] C.M. Bishop, *Neural networks for pattern recognition*, Clarendon Press, Oxford, 1996.
[23] S. Kumar, *Neural networkw*, Tsinghua University Press, Beijing, 2006.
[24] R.O. Duda, P.E. Hart and D.G. Stork, *Pattern classification (Second Edition)*, China Machine Press, Beijing, 2005.

[25] G.B. Huang, L. Chen and C.K. Siew, Universal approximation using incremental constructive feedforward networks with random hidden nodes, *IEEE Transactions on Neural Networks* **17**(4) (2006), 879–892.
[26] T.M. Cover and J.A. Thomas, *The elements of information theory (Second Edition)*. John Wiley and Sons, Inc., Hoboken, New Jersey, 2006.
[27] A. Frank and A. Asuncion, UCI Machine Learning Repository 2013, [http://archive.ics.uci.edu/ml].
[28] A. Verikas, A. Lipnickas and K. Malmqvist, Soft combination of neural classiers: A comparative study, *Pattern Recognition Letters* **20** (1999), 429–444.
[29] T. Li, Instance selection based on ELM and vote entropy (master's thesis). Hebei University, 2013.
[30] J.H. Zhai, T. Li, M.Y. Zhai and X.Z. Wang, Experimental research on random mapping functions in ELM algorithm, *Computer Engineering* **38**(20) (2012), 164–168.
[31] X. Liu, C. Gao and P. Li, A comparative analysis of support vector machines and extreme learning machines, *Neural Networks* **33**(9) (2012), 58–66.
[32] S. Ferrandiz and M. Boullé, Bayesian instance selection for the nearest neighbor rule, *Machine Learning* **81**(3) (2010), 229–256(28).