Information Sciences 367–368 (2016) 1066–1077

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Voting-based instance selection from large data sets with MapReduce and random weight networks

Junhai Zhai[a,b,*], Xizhao Wang[c], Xiaohe Pang[d]

[a] Key Lab. of Machine Learning and Computational Intelligence, College of Mathematics and Information Science, Hebei University, Baoding, 071002, Hebei, China
[b] College of Mathematics, Physics and Information Engineering, Zhejiang Normal University, Jinhua 321004, China
[c] College of Computer Science and Software, Shenzhen University, Shenzhen 518060, China
[d] College of Computer Science and Technology, Hebei University, Baoding, 071002, Hebei, China

ARTICLE INFO

Article history:
Received 16 May 2015
Revised 9 June 2016
Accepted 6 July 2016
Available online 7 July 2016

Keywords:
Large data sets
Instance selection
MapReduce
Random weight networks

ABSTRACT

Instance selection is an important preprocessing step in machine learning. By choosing a subset of a data set, it achieves the same performance of a machine learning algorithm as if the whole data set is used, and it enables a machine learning algorithm to be feasible for and to work effectively with large data sets. Based on voting mechanism, this paper proposes a large data sets instance selection algorithm with MapReduce and random weight networks (RWNs). Firstly, the proposed algorithm employs the Map of MapReduce to partition the large data sets into some small subsets, and deploys them to different cloud computing nodes. Secondly, the informative instances are selected in parallel with an instance selection algorithm. Thirdly, the Reduce of MapReduce is used to collect the selected instances from different cloud computing nodes and a selected instance subset is obtained. The previous three processes are repeated p times (p is a parameter defined by the user), and p instance subsets are obtained. Finally, the voting method is used to select the most informative instances from the p subsets. The random weight network classifier is trained with the selected instance subset, and the testing accuracy is verified on the testing set. The proposed algorithm is experimentally compared with three state-of-the-art approaches which are CNN, ENN and RNN. The experimental results show that the proposed algorithm is effective and efficient.

1. Introduction

Instance selection also named sample selection is to select a small representative subset from original data set by removing the redundant instances. In the framework of classification, the purpose of instance selection is to reduce computational complexity of classification algorithms with little or no performance deterioration. Since Hart's seminal work (i.e. CNN) [20], 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 [7]. 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

* Corresponding author.
E-mail address: mczjh@126.com (J. Zhai).

http://dx.doi.org/10.1016/j.ins.2016.07.026
0020-0255/

be misclassified by their neighbors, and thus will be retained [44]. The reduced nearest neighbor (RNN) rule proposed by Gates [17] 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 [36] improves CNN and RNN by ensuring that a MCS can be found. SNN is much more complex and its 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 [11]. The editing nearest neighbor (ENN) rule proposed by Wilson [44] 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 [7]. The reachable set depends on the distances between each instance and its nearest enemy, and the coverage set of every instance is a list of its associates.

Recently, some new instance selection algorithms were developed by different authors. Nikolaidis et al. proposed a class boundary preserving algorithm [32], which discards instances near center and retains a suitable number of instances near border. 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 [14]. The authors made the point that patterns further down the chain are close to the classification boundary. Li and Maguire presented a critical pattern selection algorithm by considering local geometrical and statistical information [27]. This algorithm selects both border and edge patterns from the data set. Based on the concept of classifier ensemble, César et al. proposed an instance selection algorithm with linear complexity [8]. Based on the divide-and-conquer principle, Nicolía et al. proposed an instance selection algorithm dealing with the class-imbalance problem [31]. In order to deal with the classification problems of large data sets, Triguero et al. proposed MapReduce-based framework for nearest neighbor classifier [39]. Based on gamma evaluator [33], Onan proposed a fuzzy-rough instance selection method for nearest neighbor classifier. This method fuzzifies conditional attributes and decision attributes, the fuzzification will result in high computational complexity. From meta-learning perspective, Leyva [26] defined a set of data-complexity measures, and applied these measures to instance selection. This is a new technical route for instance selection, but as a preprocessing step, this kind of approaches must build the meta-data, which is very difficult in some cases. Based on outlier pattern analysis and prediction, Lin et al. [29] proposed an approach for detecting the representative instances from large data sets. Based on hyperrectangle clustering, Hamidzadeh et al. [19] proposed an instance reduction method, which removes non-border(interior) instances and keeps the ones on border and near border.

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. Motivated by the idea of MapReduce [12] and voting mechanism [25], we propose an instance selection algorithm named MRVIS (MapReduce and Voting based Instance Selection) which can deal with the problems mentioned above. MRVIS consists of four steps. Firstly, the Map of MapReduce is employed to partition the large data set into some small subsets which are deployed to different cloud computing nodes. Secondly, the informative instances are selected in parallel with an instance selection algorithm (in this paper, we use the CNN for selecting the informative instances, actually arbitrary instance selection algorithm can be used). Thirdly, the Reduce of MapReduce is used to collect the selected instances from different cloud computing nodes and an instance subset is obtained. The previous three steps are repeated p times (p is a parameter defined by the user), and p instance subsets are obtained. Finally, the voting method is used to select the most informative instances from the p subsets. In this paper, the RWNs [38] also named RVFLNs (Random Vector Functional Link Networks) networks (the case of no direct link from input layer to output layer) [23,34,35] are used as classifier to test the quality of the selected instances due to their fast learning speed and good generalization ability.

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

2. Preliminaries

MapReduce and random weight networks are briefly reviewed in this section.

2.1. MapReduce

MapReduce is a simple model for distributed computing that abstracts away many difficulties in parallelizing data management operations across a cluster of commodity machines [12]. MapReduce reduces many complex tasks such as data partitioning, scheduling tasks across many machines, handling machine failures, and performing inter-machine communication. Since it is simple and easy to use, MapReduce has been successfully applied in many fields, such as machine learning, biological information processing, prediction and forecast, etc. By only using the information of protein sequences, You et al. [47] propose a novel MapReduce-based parallel support vector machine for large-scale predicting protein-protein interactions. Wu et al. [45] propose a MapReduce-based algorithm to mining event association rules in large scale distributed systems, and applied the proposed algorithm to detect events, filter irrelative events, and discover their temporal correlations. In [46], the authors discussed challenges that need to be addressed to mitigate DDoS(Distributed Denial-of-Service)

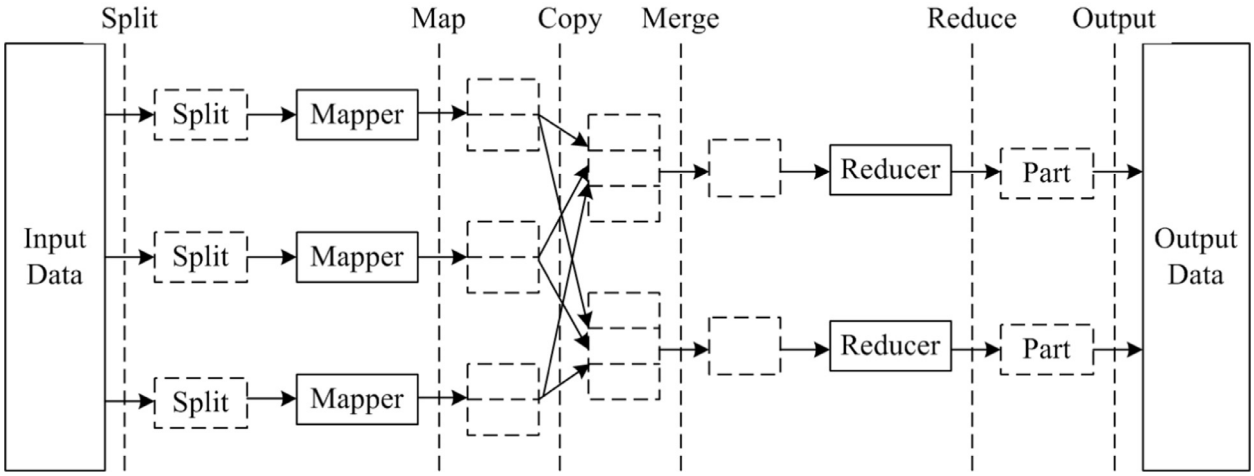

Fig. 1. Data manipulation processes of MapReduce.

attached in SDN(Software-Defined Wetworking) with MapReduce. This work can help understand how to make full use of SDN's advantages to defeat DDoS attacks in cloud computing environments and how to prevent SDN itself from becoming a victim of DDoS attacks. Regarding the problem of resource optimization in MapReduce, Li et al. [28] proposed two online dynamic resource allocation algorithms for cloud computing system. The proposed algorithm can adjust the resource allocation dynamically based on the updated information of the actual task executions. Ding et al. [13] proposed a hierarchical coevolutionary MapReduce based knowledge reduction algorithm for big data analysis. Based on frequent pattern mining, Bechini et al. [5] proposed a MapReduce associative classifier. Based on the MapReduce model, Guo et al. [16] proposed a community discovery algorithm, which can discover communities in large scale social networks efficiently and accurately. Wang [41] edited a special issue which focuses on discussing the problem of how to use uncertainty to learn with big data. Fig. 1 shows the data manipulation processes of MapReduce.

MapReduce consists of two phases: map and reduce in which users specify the computation, and the underlying runtime system automatically parallelizes the computation across large-scale clusters of machines [12]. Specifically, the computation takes a set of input key/value pairs, and produces a set of output key/value pairs. The map function written by the user takes an input pair and produces a set of intermediate key/value pairs. The MapReduce library groups together all intermediate values associated with the same intermediate key i and passes them to the reduce function. The reduce function also written by the user accepts an intermediate key i and a set of values for that key. It merges these values together to form a smaller set of values. The map and reduce functions can be written as follows.

$$map(k_1, v_1) \rightarrow list(k_2, v_2),$$
$$reduce(k_2, list(v_2)) \rightarrow list(v_2).$$

2.2. Random weight networks (RWN)

RWNs were firstly proposed by Schmidt et al. in 1992 for training single-hidden layer feed-forward neural networks (SLFNs) [38]. At the same time, a very similar model named random vector functional-link networks (RVFLNs) was independently proposed by Pao et al. [34]. The learning and generalization characteristics of the RVFLNs are investigated in [35], and the approximation capability of the RVFLNs is studied in [23]. The only difference between RWNs and RVFLNs is that there are direct links in RVFLNs from input layer to output layer.

In RWNs, the input weights and the hidden layer biases can be chosen randomly, the output weights can be analytically determined with Moore–Penrose generalized inverse [18] of the hidden layer output matrix. Unlike other gradient-descent based learning algorithms (such as back propagation algorithm [6]), RWNs does not require iterative techniques to adjust input weights and hidden layer biases during training process. By randomly initializing input weights and hidden layer bias [48], RWNs can overcome many drawbacks of the traditional gradient-based learning algorithms such as local minimal and low learning speed. Due to the fast learning speed, good generalization and approximation ability, RWNs and RVFLNs have received considerable attentions. For example, He et al. applied random weight network to fuzzy nonlinear regression analysis [21,22], in which the authors derived a new output-layer weight updating scheme based on the triangular fuzzy number input-output data set [21], and revealed some feasible guidelines to the applications of uncertainty in multiple criteria decision making techniques [22]. An iterative learning algorithm for RWNs was proposed in [10], which can train RWNs with large scale data sets. A fast decorrelated neural network ensemble method for RWNs was proposed in [1], which can overcome the drawbacks of the negative correlation ensemble learning. A local learning algorithm and a probabilistic learning algorithm for RWNs were proposed in [51] and [9] respectively, both of which can improve the robustness

of learning system. In [37], a distributed learning algorithm for RVFLNs was proposed to handle the problem of large scale learning. Lu et al. [30] discussed the computational problem of Moore–Penrose inverse, and proposed two effective methods for computing Moore–Penrose inverse. Recently, Wang et al. [42,43] studied the relationship between generalization abilities and uncertainties of base classifiers in ensemble learning, and obtained very valuable conclusion: the classifier with higher uncertainty outputs has better generalization for complex boundary problems. Arqub et al. [2] conducted a further investigation on modeling problem of uncertainty of dynamic systems, and some solutions to the problem were given in [3,4]. Zhang and Suganthan presented a comprehensive evaluation and an excellent survey on random vector functional link networks in [49] and [50] respectively.

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

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

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 initialized, and $\beta_j = (\beta_{j1}, \beta_{j2}, \cdots, \beta_{jm})^T$ is the weight vector connecting the jth hidden node with the output nodes. The parameters $\beta_j (j = 1, 2, \cdots, m)$ can be estimated by least-square fitting with the given training data set D, i.e. , satisfying

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

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

$$H\beta = Y, \tag{3}$$

where

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

$$\beta = (\beta_1^T, \cdots, \beta_m^T), \tag{5}$$

and

$$Y = (y_1^T, \cdots, y_n^T). \tag{6}$$

H is the hidden layer output matrix of the network, where the jth column of $H$is the jth hidden node's 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). Approximating Eq. (3) using the least-square fitting is to solve the following equation:

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

The smallest norm least-squares solution to (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 RWN algorithm [38] is presented as follows.

3. The proposed instance selection algorithm

In this section, we firstly present the idea of the proposed algorithm, and then present the proposed algorithm. The idea of the proposed algorithm is simple, it contains four phases. In the first phase, the large data set is partitioned into several small subsets with Map of MapReduce, and then these subsets are deployed to different cloud computing nodes. In the second phase, the informative instances are selected from the deployed subset in parallel with an instance selection algorithm (in this paper, we use CNN algorithm which is simple and easy to implemented). In the third phase, the Reduce of MapReduce is used to merge the selected instances from different cloud computing nodes, and an instance subset is obtained. The previous three phases are repeated p times to generate p instance subsets. In the fourth phase, we cast votes for the instances in the p instance subsets. If an instance x in one of p subsets, then x receives a vote, the number of votes of an instance x is denoted by $vote(x)$. Obviously, the more votes a selected instance received, the more important this instance is. The final selected instances are the ones whose number of votes is greater than or equal to a user predefined threshold

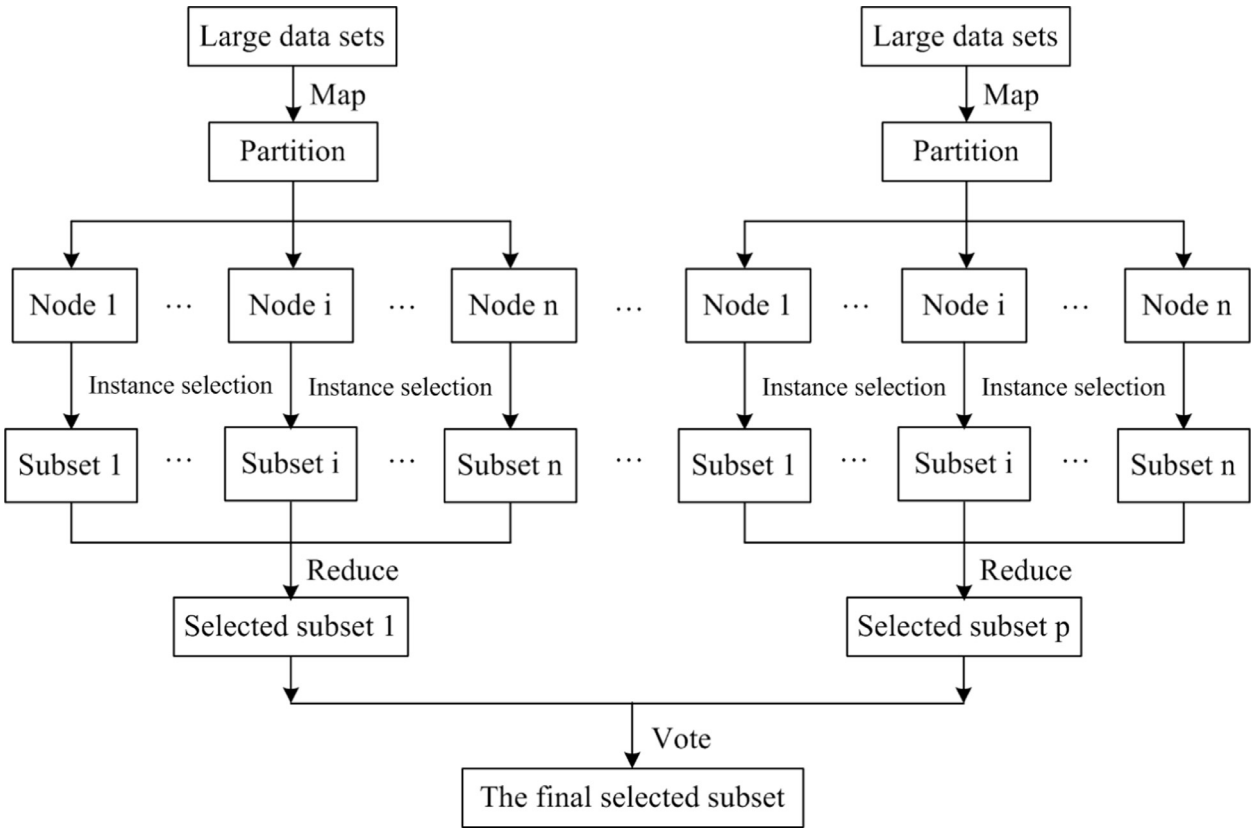

Fig. 2. The idea of the proposed algorithm.

Algorithm 1: RWN 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 biases b_j,$j = 1, 2, \cdots, m$;
2 Calculate the hidden layer output matrix H;
3 Calculate output weights matrix $\hat{\beta} = H^{\dagger}Y$.

λ. The idea of the proposed algorithm is illustrated in Fig. 2. Fig. 3 shows an example of the proposed algorithm MRVIS, the data set used in this example includes 50 instances, and $p = 3$, $\lambda = 3$.

Given a training data set $D = \{(x_i, y_i)|x_i \in R^d, y_i \in R^k, i = 1, 2, \cdots, n\}$, the proposed algorithm is described in Algorithm 2.

It is well known that the computational time complexities of CNN, RNN, and ENN are $O(ksn^2)$, $O(ksn^2)$ and $O(kn^2)$ respectively [24], where n is the number of instances of data set, k is the number of nearest neighbors, and s is the number of the selected instances. In order to compare the proposed algorithm MRVIS with CNN, RNN, and ENN, a simple theoretical analysis of the computational time complexity of the proposed algorithm MRVIS is presented.

The algorithm MRVIS includes 14 steps. It is obvious that the computational time complexity of the step 1, the for loop (steps 6–9), the if statement (steps 11–13), and the step 14 are all $O(1)$. Since the number of instances of data set is n, it is easy to obtain the computational time complexity of step 2 is $O(n)$. If t is the number of computing nodes, and the instance selection algorithm used in MRVIS is CNN, then the computational time complexities of the step 3 and step 4 are $O(\frac{1}{t}ksn^2)$ and $O(ts)$ respectively.

Accordingly, the computational time complexity of algorithm MRVIS is $3 \times O(1) + O(n) + O(\frac{1}{t}ksn^2) + O(ts)$. Obviously, the computational time complexity of algorithm MRVIS is $O(\frac{1}{t}ksn^2)$ in the worst situation. We summarize the results in Table 1. From the Table 1, it is clear that the computational time complexity of MRVIS is the lowest compared with CNN, RNN, and ENN.

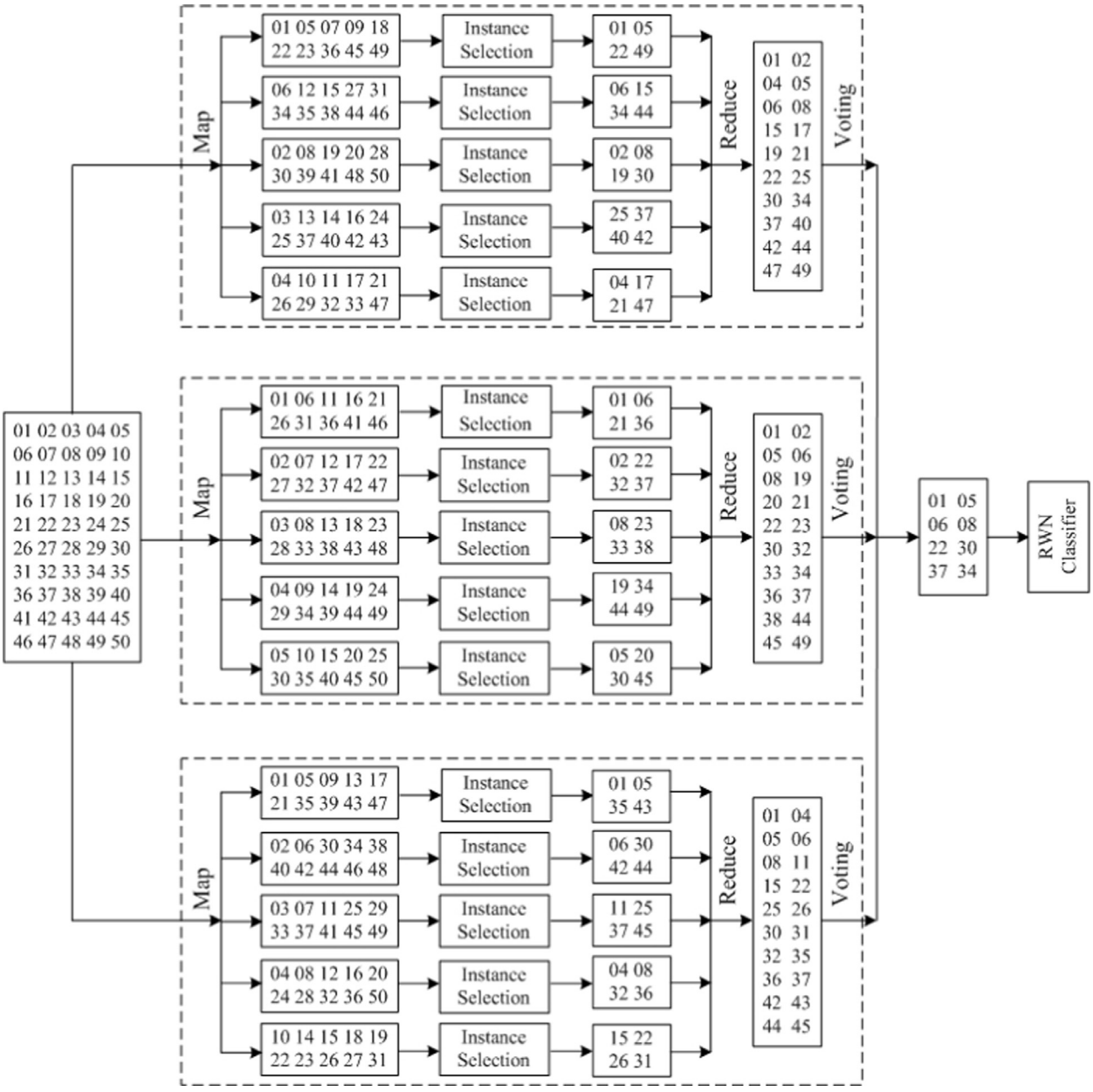

Fig. 3. An example of the proposed algorithm MRVIS with $p = 3$, $\lambda = 3$.

The key issue of the implementation of the proposed algorithm is the design of the mapper function and the reducer function of MapReduce. In our implementation, we select the important instances in parallel with CNN, and vote in parallel for the selected instances in cloud computing platform. We present the design of the mapper function and the reducer function for selecting instance with CNN in Algorithms 3 and 4 respectively. The design of the mapper function and the reducer function for voting for instances can be similarly obtained.

In Algorithms 3 and 4, key is the class label, value is all attributes. HDFS is the Hadoop Distributed File System.

4. Experimental results and analysis

The effectiveness of our proposed method is verified through numerical experiments in a cloud computing platform with 6 nodes, the configuration of the cloud computing platform is given in Table 2, the configuration of nodes of the cloud computing platform is given in Table 3.

Algorithm 2: MRVIS: MapReduce and Voting based Instance Selection.

Input: Training data set $D = \{(x_i, y_i)|x_i \in R^d, y_i \in R^k, i = 1, 2, \cdots, n\}$, the number of iteration p, and the threshold of votes λ.
Output: The selected instance subset S.

Initialize $S = \phi$;
Partition the large data set into several small subsets with map mechanism of MapReduce, and deploy them to different cloud computing nodes;
Parallelly select the informative instances from the subsets in different cloud computing nodes with an arbitrary instance selection algorithm;
Merge the selected instances by different cloud computing nodes, and obtain a selected instance subset;
Repeat steps 2–4 p times, and obtain p selected subsets $S_1, S_2, \cdots, S_p$;
for $(i = 1; i \leq p; i = i + 1)$ **do**
  if $(x \in S_i)$ **then**
    $vote(x) = vote(x) + 1$;
  end
end
if $(vote(x) \geq \lambda)$ **then**
  $S = S \cup \{x\}$;
end
Return S.

Table 1
Comparison of computing time complexity of the 4 algorithms.

Algorithms	CNN	RNN	ENN	MRVIS
Complexity	$O(ksn^2)$	$O(ksn^2)$	$O(kn^2)$	$O(\frac{1}{t}ksn^2)$

Algorithm 3: The mapper function for selecting instances with CNN.

Input: key, value, context.
Output: The selected instance subset S.

Initialize T with the data that mapper received;
Randomly pick an instance x_i from T;
$S = x_i$;
$T = T - x_i$;
repeat
  $append = FALSE$;
  for $(each\ x \in T)$ **do**
    find instance s in S, such that $d(x, s) = \min_{s_j \in S} d(x, s_j)$;
    if $(Class(x) \neq Class(s))$ **then**
      $S = S \cup x$;
      $T = T - x$;
      $append = TURE$;
    end
  end
until *append=FALSE*;
return S.

Algorithm 4: The reducer function for selecting instances with CNN.

Input: key, value, context.
Output: key, value.

for (*all instances received form each mapper*) **do**
  sort the pair of key and value;
  output the results to HDFS;
end

Table 2
The configuration of the cloud computing platform.

Items	Configuration
Operating System	Ubuntu 13.04
Hadoop	Hadoop 0.20.2
JDK	JDK-7u71-linux-i586
Eclipse	Eclipse-java-luna-SR1-linux

Table 3
The configuration of the nodes of the cloud computing platform.

Items	Configuration
CPU	Inter Xeon E5-4603 with two cores, 2.0GZ
Memory	8G
Network Card	Broadcom 5720 QP 1Gb
Hard Disk	1TB

Table 4
The basic information of the 8 data sets.

Data sets	#Instances	#Attributes	#Classes
Banana	5300	2	2
Cloud	10000	2	2
Gaussian	20000	2	2
Shuttle	58000	9	7
Artificial	250000	10	2
cod_rn	487565	8	2
Poker	1025010	10	10
Susy	5000000	17	2

Two experiments are conducted on 8 data sets including 2 artificial data sets and 6 UCI data sets [15]. The first experiment is to determine a suitable range of the random weights and the suitable values of parameters p and λ. The second experiment is to compare the proposed algorithm MRVIS with three state-of-the-art approaches CNN, ENN and RNN on three aspects: the number of selected instances, the condensed ratio, and testing accuracy. There are 4 small data sets and 4 large data sets in the 8 selected data sets, the largest data set includes five hundred million instances with 17 attributes and two classes. For each data set, the 10-fold cross-validation are run 10 times. The experimental results are the average of the 10 outputs. The basic information of the 8 data sets is listed in Table 4.

The first artificial data set is two-dimensional cloud data with two equal priori probable classes [40]. The class ω_1 is the mixture of three different Gaussian distributions:

$$p(x|\omega_1) = \frac{1}{2}\left(\frac{p_1(x)}{2} + \frac{p_2(x)}{2} + p_3(x)\right), \tag{9}$$

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

$$p_i(x) = \frac{1}{2\pi\sigma_{ix_1}\sigma_{ix_2}} \times \exp\left(-\frac{(x_1-\mu_{ix_1})^2}{2\sigma_{ix_1}^2} - \frac{(x_2-\mu_{ix_2})^2}{2\sigma_{ix_2}^2}\right), \tag{10}$$

where μ_{ix_1} and μ_{ix_2} are the means of x_1 and x_2 of the ith Gaussian component, σ_{ix_1} and σ_{ix_2} are the corresponding standard deviations.

The class ω_2 is a single Gaussian distribution:

$$p(x|\omega_2) = \frac{1}{2\pi}\exp\left(-\frac{x_1^2+x_2^2}{2}\right). \tag{11}$$

The second artificial data set is a two-dimensional Gaussian data with two classes $\omega_i (i = 1, 2)$, the distribution of ω_i is

$$p(x|\omega_i) \sim N(\mu_i, \Sigma_i), \tag{12}$$

where $\mu_1 = (0.1597, 1.3541)^T$, $\mu_2 = (1.1597, 1.4541)^T$, $\Sigma_1 = \begin{bmatrix}0.1726 & 0.0912\\ 0.0912 & 0.1020\end{bmatrix}$, and $\Sigma_2 = \begin{bmatrix}0.1726 & 0.0912\\ 0.0912 & 0.1020\end{bmatrix}$.

4.1. Experiment 1: To determine a suitable range of the random weights and the suitable values of parameters p *and* λ

For random weight networks, it is important to generate random weights with a uniform distribution within a suitable range, which is usually set to the interval $[-1, +1]$. However, there are no theoretical or experimental analysis on the ra-

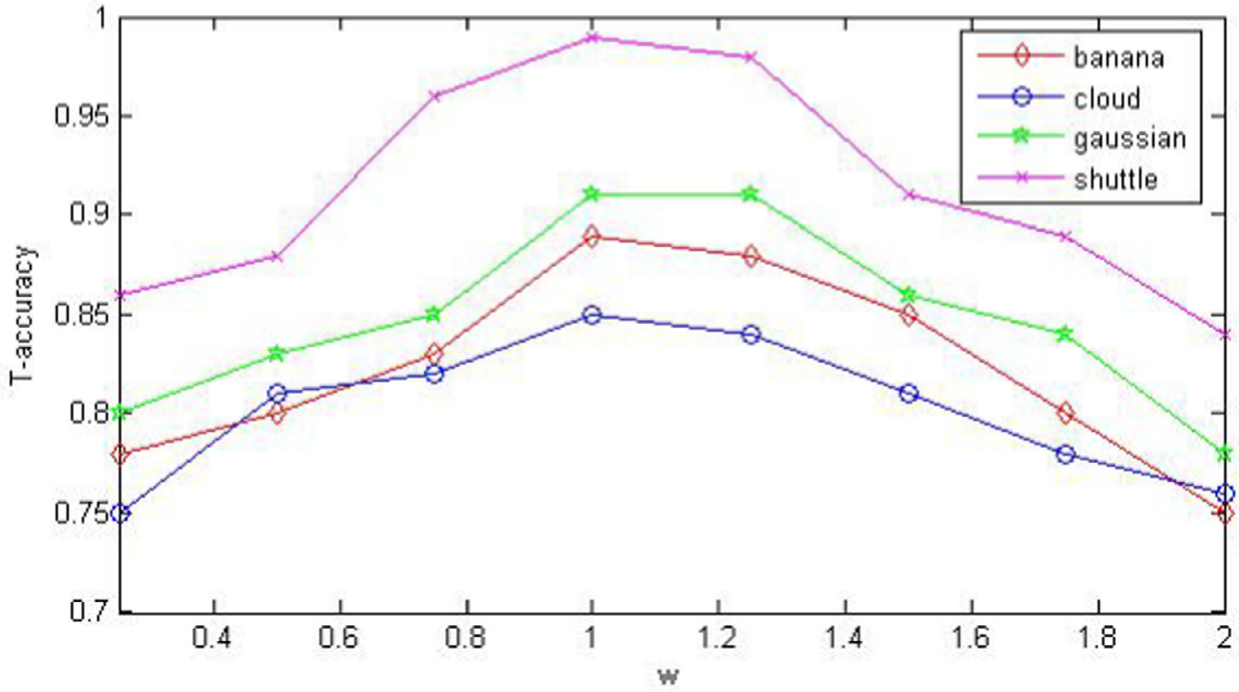

Fig. 4. The relationship between T-accuracy and the w on 4 small data sets.

Table 5
The influence of the two parameters on the performance of the proposed algorithm.

Artificial			cod_rn			Poker			Susy		
p	λ	T-accuracy	p	λ	T-accuracy	p	λ	T-accuracy	p	λ	T-accuracy
1	1	0.5521	1	1	0.8923	1	1	0.8209	1	1	0.8012
3	2	0.6019	3	2	0.9228	3	2	0.8927	3	2	0.8609
5	3	0.6266	5	3	0.9409	5	3	0.9205	3	2	0.8851
6	4	0.6272	6	4	0.9409	7	4	0.9205	3	2	0.8927

tionality of this setting in the literature. In experiment 1, we conduct an experimental analysis on the 8 selected data sets, and conclude that in our experiments this setting is suitable. Furthermore, we conjecture that this setting is reasonable for most applications. In experiment 2, the random weighting are generated in this way.

In our work, we use random weight networks as classifiers to test the selected subsets of instances. In order to determine the suitable range of the random weights, for each data set, we generate the random weights with a uniform distribution within interval $[-w, +w]$, where w is a positive real number. The w is initialized to 0.25 and increased by 0.25 each time until it is equal to 2.0. With w set to different values we record the average testing accuracies (denoted by T-accuracy) of the RWNs with fixed structure. The experimental results on 4 small data sets and on 4 large data sets are shown in Figs. 4 and 5. From Figs. 4 and 5, we can see that the value of w does affect the performance of the RWNs, and the RWNs' performance are poor when the value of w is too small or too large, for example, when $w = 0.25$ or $w = 2.00$. For all data sets, the optimal performance are taken when $w = 1$. Accordingly, we experimentally conclude that the suitable range of random weights is $[-1, +1]$.

In the proposed algorithm MRVIS, there are two user defined parameters p and λ. Generally, the parameter λ should satisfy $\lambda > 0.5$, which means that the selected instances should receive more than half the votes. Regarding the parameter p, the experimental results show that the testing accuracy (the T-accuracy in the following Table 5) achieves the largest values on most data sets when p is equal to 5 or 6. In experiment 2, we set $p = 6$ and $\lambda = 4$. The influence of the two parameters on the performance of the proposed algorithm MRVIS is listed in Table 5.

4.2. Experiment 2: The comparison between the proposed algorithm with CNN, ENN and RNN

In experiment 2, we compare the proposed algorithm MRVIS with the three state-of-the-art approaches CNN, ENN and RNN. On the 4 small data sets, we compare MRVIS with CNN, ENN and RNN on three aspects: the number of selected instances, the condensed ratio, and testing accuracy. The comparisons of performances are listed inTables 6–9. While on 4

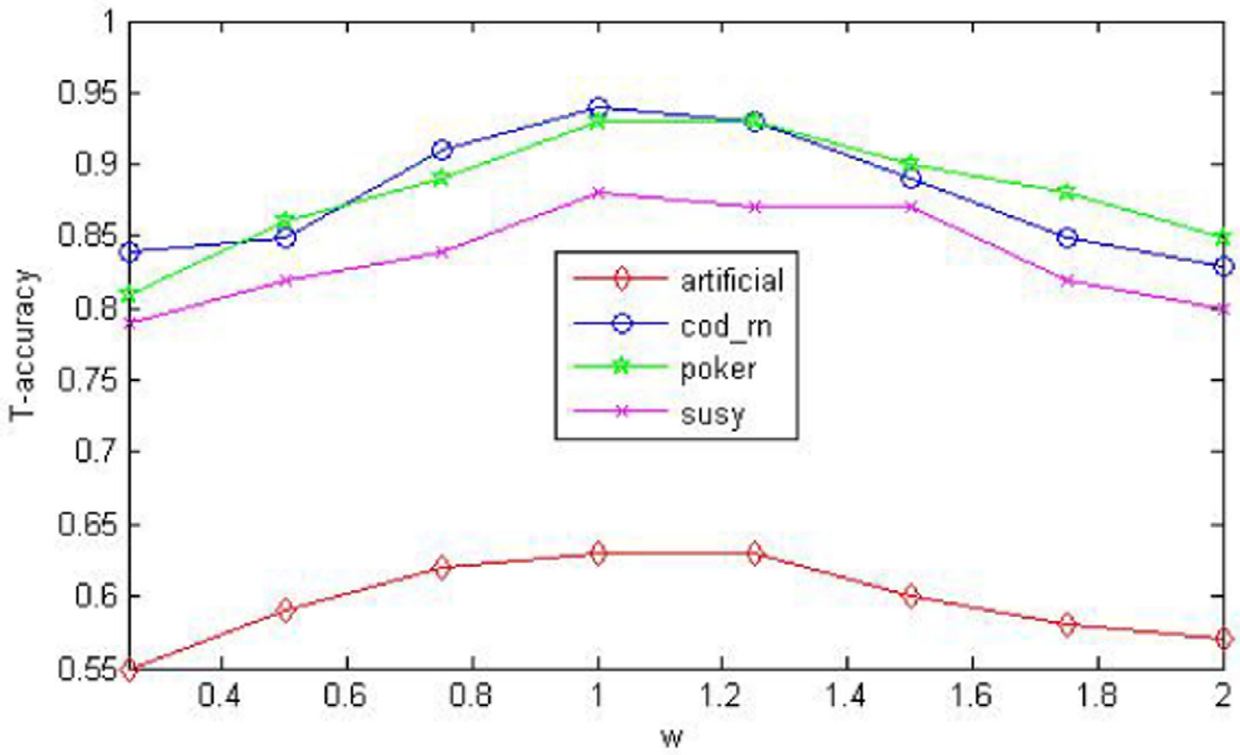

Fig. 5. The relationship between T-accuracy and the w on 4 large data sets.

Table 6
The experimental results on data set banana.

Algorithms	S-number	S-ratio	T-accuracy
CNN	752	4.6542	0.8854
ENN	2893	1.2098	0.8945
RNN	739	4.7361	0.8791
MRVIS	433	8.0831	0.8796

Table 7
The experimental results on data set cloud.

Algorithms	S-number	S-ratio	T-accuracy
CNN	1675	3.9797	0.8403
ENN	5492	1.2138	0.8942
RNN	1599	4.1689	0.8399
MRVIS	785	8.4917	0.8418

Table 8
The experimental results on data setGaussian .

Algorithms	S-number	S-ratio	T-accuracy
CNN	2585	5.1579	0.9172
ENN	11843	1.1258	0.9418
RNN	2540	5.2492	0.9171
MRVIS	1647	8.0953	0.9129

Table 9
The experimental results on data set shuttle.

Algorithms	S-number	S-ratio	T-accuracy
CNN	564	68.5567	0.9978
ENN	37845	1.0270	0.9952
RNN	541	71.4713	0.9974
MRVIS	397	97.3955	0.9854

Table 10
The experimental results on data set artificial.

Algorithms	S-number	S-ratio	T-accuracy	CPU time
CNN	91754	1.8164	0.6048	18415
ENN	-	-	-	-
RNN	-	-	-	-
MRVIS	41786	3.9886	0.6272	451

Table 11
The experimental results on data set cod_rn.

Algorithms	S-number	S-ratio	T-accuracy	CPU time
CNN	39813	8.1810	0.9389	24967
ENN	-	-	-	-
RNN	-	-	-	-
MRVIS	21867	14.8950	0.9409	768

Table 12
The experimental results on data set poker.

Algorithms	S-number	S-ratio	T-accuracy	CPU time
CNN	-	-	-	-
ENN	-	-	-	-
RNN	-	-	-	-
MRVIS	61242	11.1580	0.9205	4621

Table 13
The experimental results on data set susy.

Algorithms	S-number	S-ratio	T-accuracy	CPU time
CNN	-	-	-	-
ENN	-	-	-	-
RNN	-	-	-	-
MRVIS	268415	12.4186	0.8827	15682

large data sets, besides the mentioned three aspects, we also compared the CPU time. The experimental results are given inTables 10–13. In the Tables 6–13, S-number denotes the number of selected instances, C-ratio denotes the condensed ratio, T-accuracy denotes the testing accuracy respectively, while in the Tables 10–13, the symbol "-" means that the results cannot be obtained. Compared with CNN, ENN, and RNN, from the experimental results listed in Tables 6–13, we can find that our algorithm removes much more instances while obtaining the similar accuracies. What's more, CNN do not work out results on the two largest data sets: poker and susy, ENN and RNN do not work out results on all four largest data sets, while our proposed algorithm can work well on all four large data sets.

5. Conclusions

Based on MapReduce and voting mechanism, this paper proposes a large data set instance selection algorithm named MRVIS, which is practicable on large data sets, while some classic algorithms (e.g. CNN, ENN, RNN) are impracticable. Furthermore, the proposed algorithm has three major advantages: (1) fast learning speed, (2) high condensed ratio, and (3) no limitation on the instance selection algorithm used in MRVIS. The fast learning speed is due to the parallelization of selecting informative instances, the high condensed ratio is achieved by voting mechanism. The experimental results have verified that the proposed algorithm is much more feasible and effective than the three state-of-the-art approaches CNN, ENN and RNN. In our future works, (1) we will introduce a regularization term into optimal problem (7) and investigate its impact on the performance of the proposed algorithm. (2) We will theoretically analyze the rationality of the setting of random weights.

Acknowledgment

This research is supported by Basic Research Project of Knowledge Innovation Program in Shenzhen (JCYJ20150324140036825), by National Natural Science Foundations of China (71371063), by 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] M. Alhamdoosh, D.H. Wang, Fast decorrelated neural network ensembles with random weights, Inf. Sci. 264 (6) (2014) 104–117.
[2] O.A. Arqub, Z. Abo-Hammour, Numerical solution of systems of second-order boundary value problems using continuous genetic algorithm, Inf. Sci. 279 (2014) 396–415.
[3] O.A. Arqub, M. AL-Smadi, S. Momani, et al., Numerical solutions of fuzzy differential equations using reproducing kernel Hilbert space method, Soft Comput. (2015), doi:10.1007/s00500-015-1707-4.
[4] O.A. Arqub, Adaptation of reproducing kernel algorithm for solving fuzzy Fredholm-Volterra integrodifferential equations, Neural Comput. Appl. (2015), doi:10.1007/s00521-015-2110-x.
[5] A. Bechini, F. Marcelloni, A. Segatori, A MapReduce solution for associative classification of big data, Inf. Sci. 332 (2016) 33–55.
[6] C.M. Bishop, Neural Networks for Pattern Recognition, Clarendon Press, Oxford, 1996.
[7] B. Brighton, C. Mellish, Advances in instance selection for instance-based learning algorithms, Data Min. Knowl. Discov. 6 (2) (2002) 153–172.
[8] G.O. César, H.G.A. de, G.P. Nicolás, Democratic instance selection: a linear complexity instance selection algorithm based on classifier ensemble concepts, Artif. Intell. 174 (5–6) (2010) 410–441.
[9] F.L. Cao, H.L. Ye, D.H. Wang, A probabilistic learning algorithm for robust modeling using neural networks with random weights, Inf. Sci. 313 (2015) 62–78.
[10] F.L. Cao, D.H. Wang, H.Y. Zhu, et al., An iterative learning algorithm for feedforward neural networks with random weights, Inf. Sci. 328 (2016) 546–557.
[11] B.V. Dasarathy, Minimal consistent set (MCS) identification for optimal nearest neighbor decision systems design, IEEE Trans. Syst. Man Cybern. 24 (1) (1994) 511–517.
[12] J. Dean, S. Ghemawat, MapReduce: Simplified data processing on large clusters, Commun. ACM 51 (1) (2008) 107–113.
[13] W.P. Ding, J.H. Wang, J.D. Wang, A hierarchical-coevolutionary-MapReduce-based knowledge reduction algorithm with robust ensemble Pareto equilibrium, Inf. Sci. 342 (2016) 153–175.
[14] H.A. Fayed, A.F. Atiya, A novel template reduction approach for the k-nearest neighbor method, IEEE Trans. Neural Netw. 20 (5) (2009) 890–896.
[15] A. Frank, A. Asuncion, UCI Machine Learning Repository, 2013,. http://archive.ics.uci.edu/ml.
[16] K. Guo, W. Guo, Y. Chen, et al., Community discovery by propagating local and global information based on the MapReduce model, Inf. Sci. 323 (2015) 73–93.
[17] G.W. Gates, The reduced nearest neighbor rule, IEEE Trans. Inf. Theory 18 (3) (1972) 431–433.
[18] G.H. Golub, C.F.V. Loan, Matrix Computations, 3rd edition, The Johns Hopkins University Press, Baltimore, MD, 1996.
[19] J. Hamidzadeh, R. Monsefi, H.S. Yazdi, IRAHC: instance reduction algorithm using hyperrectangle clustering, Pattern Recognit. 48 (5) (2015) 1878–1889.
[20] P. Hart, The condensed nearest neighbor rule, IEEE Trans. Inf. Theory 14 (5) (1967) 515–516.
[21] Y.L. He, X.Z. Wang, J.Z.X. Huang, Fuzzy nonlinear regression analysis using a random weight network, Inf. Sci. (2016a), doi:10.1016/j.ins.2016.01.037. In press.
[22] Y.L. He, X.Z. Wang, J.Z.X. Huang, Recent advances in multiple criteria decision making techniques, Int. J. Mach. Learn. Cybern. (2016b), doi:10.1007/s13042-015-0490-y. In press.
[23] B. Igelnik, Y.H. Pao, Stochastic choice of basis functions in adaptive function approximation and the functional-link net, IEEE Trans. Neural Netw. 6 (6) (1995) 1320–1329.
[24] N. Jankowski, M. Grochowski, Comparison of instances selection algorithms i. algorithms survey, Curr. Gastroenterol. Rep. 10 (1) (2008) 0937–0942.
[25] J. Kittler, M. Hatef, R.P.W. Duin, et al., On combining classifiers, IEEE Trans. Pattern Anal. Mach. Intell. 20 (3) (1998) 226–239.
[26] E. Leyva, A. González, R. Pérez, A set of complexity measures designed for applying meta-learning to instance selection, IEEE Trans. Knowl. Data Eng. 27 (2) (2015) 354–367.
[27] Y.H. Li, L. Maguire, Selecting critical patterns based on local geometrical and statistical information, IEEE Trans. Pattern Anal. Mach. Intell. 33 (6) (2011) 1189–1201.
[28] J. Li, M. Qiu, Z. Ming, et al., Online optimization for scheduling preemptable tasks on IaaS cloud systems, J. Parallel Distrib. Comput. 72 (5) (2012) 666–677.
[29] W.C. Lin, C.F. Tsai, S.W. Ke, et al., Learning to detect representative data for large scale instance selection, J. Syst. Softw. 106 (2015) 1–8.
[30] S.X. Lu, X.Z. Wang, G.Q. Zhang, et al., Effective algorithms of the Moore–Penrose inverse matrices for extreme learning machine, Intell. Data Anal. 19 (4) (2015) 743–760.
[31] G.P. Nicolía, P.R. Javier, H.G.A. de, OligoIS: scalable instance selection for class-imbalanced data sets, IEEE Trans. Cybern. 43 (1) (2013) 332–346.
[32] K. Nikolaidis, J.Y. Goulermas, Q.H. Wu, A class boundary preserving algorithm for data condensation, Pattern Recognit. 44 (3) (2011) 704–715.
[33] A. Onan, A fuzzy-rough nearest neighbor classifier combined with consistency-based subset evaluation and instance selection for automated diagnosis of breast cancer, Expert Syst. Appl. 42 (20) (2015) 6844–6852.
[34] Y.H. Pao, S.M. Phillips, D.J. Sobajic, Neural-net computing and the intelligent control of systems, Int. J. Control 56 (2) (1992) 263–289.
[35] Y.H. Pao, G.H. Park, D.J. Sobajic, Learning and generalization characteristics of the random vector functional-link net, Neurocomputing 6 (94) (1994) 163–180.
[36] G.L. Ritter, H.B. Woodruff, S.R. Lowry, An algorithm for a selective nearest neighbor decision rule, IEEE Trans. Inf. Theory 21 (6) (1975) 665–669.
[37] S. Scardapane, D.H. Wang, M. Panella, et al., Distributed learning for random vector functional-link networks, Inf. Sci. 301 (2015) 271–284.
[38] W.F. Schmidt, M.A. Kraaijveld, R.P.W. Duin, Feed forward neural networks with random weights, in: Proceedings of 11th IAPR International Conference on Pattern Recognition Methodology and Systems, 2, 1992, pp. 1–4.
[39] I. Triguero, D. Peralta, J. Bacardit, et al., MRPR: a MapReduce solution for prototype reduction in big data classification, Neurocomputing 150(Part A) (2015) 331–345.
[40] A. Verikas, A. Lipnickas, K. Malmqvist, Soft combination of neural classiers: a comparative study, Pattern Recognit. Lett. 20 (1999) 429–444.
[41] X.Z. Wang, Uncertainty in learning from big data-editorial, J. Intell. Fuzzy Syst. 28 (5) (2015) 2329–2330.
[42] X.Z. Wang, H.J. Xing, Y. Li, et al., A study on relationship between generalization abilities and fuzziness of base classifiers in ensemble learning, IEEE Trans. Fuzzy Syst. 23 (5) (2015a) 1638–1654.
[43] X.Z. Wang, R.A.R. Ashfaq, A.M. Fu, Fuzziness based sample categorization for classifier performance improvement, J. Intell. Fuzzy Syst. 29 (3) (2015b) 1185–1196.
[44] D.R. Wilson, T.R. Martinez, Reduction techniques for instance-based learning algorithms, Mach. Learn. 38 (3) (2000) 257–286.
[45] G. Wu, H. Zhang, M. Qiu, et al., A decentralized approach for mining event correlations in distributed system monitoring, J. Parallel Distrib. Comput. 73 (3) (2013) 330–340.
[46] Q. Yan, F. Yu, Distributed denial of service attacks in software-defined networking with cloud computing, Commun. Mag. IEEE 53 (4) (2015) 52–59.
[47] Z.H. You, J.Z. Yu, L. Zhu, et al., A mapreduce based parallel SVM for large-scale predicting protein-protein interactions, Neurocomputing 145 (18) (2014) 37–43.
[48] J.H. Zhai, H.Y. Xu, X.Z. Wang, Dynamic ensemble extreme learning machine based on sample entropy, Soft Comput. 16 (9) (2012) 1493–1502.
[49] L. Zhang, P.N. Suganthan, A comprehensive evaluation of random vector functional link networks, Inf. Sci. (2015). http://dx.doi.org/10.1016/j.ins.2015.09.025.
[50] L. Zhang, P.N. Suganthan, A survey of randomized algorithms for training neural networks, Inf. Sci. (2016). http://dx.doi.org/10.1016/j.ins.2016.01.039.
[51] J.W. Zhao, Z.H. Wang, F.L. Cao, et al., A local learning algorithm for random weights networks, Knowl.-Based Syst. 74 (2014) 159–166.