A novel ensemble method for k-nearest neighbor

Youqiang Zhang , Guo Cao , Bisheng Wang , Xuesong Li

PII: S0031-3203(18)30279-6
DOI: https://doi.org/10.1016/j.patcog.2018.08.003
Reference: PR 6632

To appear in: *Pattern Recognition*

Received date: 10 April 2018
Revised date: 19 June 2018
Accepted date: 1 August 2018

Please cite this article as: Youqiang Zhang , Guo Cao , Bisheng Wang , Xuesong Li , A novel ensemble method for k-nearest neighbor, *Pattern Recognition* (2018), doi: https://doi.org/10.1016/j.patcog.2018.08.003

Highlights:

- We proposed a weighted heterogeneous distance metric (WHDM).
- We presented WHDM and Dempster-Shafer theory based *k*NN algorithm.
- We proposed a multimodal perturbation method (RRSB) for *k*NN ensemble.
- The effectiveness of our algorithms was shown on multiple UCI data sets and a KDD data set.

Tittle: A novel ensemble method for *k*-nearest neighbor

Author order: Youqiang Zhang, Guo Cao[*], Bisheng Wang, Xuesong Li

Affiliation: *School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, PR China*

[*] Corresponding author. Tel.: +86 25 843 172 97.
E-mail address: zhangyouqiang@foxmail.com (Youqiang Zhang), caoguo@njust.edu.cn (Guo Cao)

A novel ensemble method for *k*-nearest neighbor

Abstract: In this paper, to address the issue that ensembling *k*-nearest neighbor (*k*NN) classifiers with resampling approaches cannot generate component classifiers with a large diversity, we consider ensembling *k*NN through a multimodal perturbation-based method. Since *k*NN is sensitive to the input attributes, we propose a weighted heterogeneous distance Metric (WHDM). By using a WHDM and evidence theory, a progressive *k*NN classifier is developed. Based on a progressive *k*NN, the random subspace method, attribute reduction, and Bagging, a novel algorithm termed RRSB (reduced random subspace-based Bagging) is proposed for construct ensemble classifier, which can increase the diversity of component classifiers without damaging the accuracy of the component classifiers. In detail, RRSB adopts the perturbation on the learning parameter with a weighted heterogeneous distance metric, the perturbation on the input space with random subspace and attribute reduction, the perturbation on the training data with Bagging, and the perturbation on the output target of *k* neighbors with evidence theory. In the experimental stage, the value of *k*, the different perturbations on RRSB and the ensemble size are analyzed. In addition, RRSB is compared with other multimodal perturbation-based ensemble algorithms on multiple UCI data sets and a KDD data set. The results from the experiments demonstrate the effectiveness of RRSB for *k*NN ensembling.

Keywords: Distance metric; *k*-nearest neighbor; ensemble learning; random subspace; evidence theory

1. Introduction

Ensemble learning has been a prominent topic in the field of machine learning in recent years, and it is listed as the first of four research directions in machine learning research by Dietterich [1]. To enhance the generalization performance of ensemble learning, many different approaches have been proposed for training accurate but diverse component classifiers. According to the mode of training the classifier, the typical ensemble approaches can be divided into three cases [2]:

- component classifier is trained on a different attribute subspace.
- component classifier is trained on different resampling training data.
- component classifier is trained on a data set with several different parameters.

The ensemble scheme may take into account any of the above three techniques. For example, each component classifier is trained on a randomly selected attribute space in the case of the random subspace method (RSM) [3, 4]. Ho [3] first proposed RSM and applied it in a decision tree ensemble and then investigated RSM in a *k*NN ensemble [4]. Gu et al. [5] proposed a random subspace-based sparse representation ensemble algorithm, where sparse representations in multiple subspaces are integrated into an ensemble sparse representation. Rotation forest (RoF) [6, 7] is an improved version of RSM. Genetic algorithm (GA) is also used to select a best fitting attribute space for each component classifier [8]. Bagging obtains different component classifiers through training on bootstrap sampling data [9]. Boosting is another resampling-based ensemble method, which considers the weight probability distribution of resampling at each trial [10]. The perturbation of parameters is often applied in neural network ensembles. For instance, random initial weights are used to train each neural network [11]. Gabrys and Ruta [12] used GA for selecting classifier prototypes, attribute space and combination rules simultaneously.

Unlike neural network and decision tree classifiers have many parameters, *k*NN classifier has only two parameters, i.e., the distance measure for computing the distance of a given test sample to the training samples and the number of neighbors *k*, which makes *k*NN ensembles challenging. Although Bagging has achieved great success on decision trees [13] and neural networks [14], it can hardly work well on *k*NN classifier because *k*NN is a stable classifier. As Breiman [9] pointed out that Bagging can hardly work on *k*NN because Bagging uses the bootstrap resampling technique to generate accurate but diverse component classifiers, which is effective on unstable methods such as decision tree and neural network.

Many research papers have investigated *k*NN ensembles with the aim of improving their performance. For instance, Bao et al. [15] applied multiple distance metrics to generate diverse ensemble members, where the distance metrics were treated as learning parameter perturbations. Ishii et al. [16] ensembled *k*NN by using GA to weight different distance functions. Multiple random subspaces used to obtain component *k*NN classifiers was investigated by Ho [4], which trained each *k*NN on a random attribute subset rather than on the whole attribute space. Zhou and Yu [2] used bootstrap sampling, attribute filtering and randomly configured distance metrics for *k*NN ensembles, which simultaneously employed perturbations on training data, attribute space and learning parameters. Altinçay [17] proposed GA-based multimodal perturbation for *k*NN ensembles, which uses GA to jointly estimate both the best fitting attribute subsets and learning parameters of each member classifier. Nanni and Lumini [18] proposed PSO-based multimodal perturbation for *k*NN ensembles, where RSM is used to perturb the attribute space, and PSO is adopted to perturb the learning parameters of each member classifier.

There are some issues in the above *k*NN ensemble methods: 1) Euclidean distance metrics handle the heterogeneous attributes in a simple way and do not consider the weight of different attributes; 2) Perturbation of the attribute space using RSM may damage the accuracy of component classifiers, since *k*NN is sensitive to the input attribute space; and 3) GA- and PSO-based multimodal perturbation for *k*NN ensembles require a large computation cost.

In this work, to address the above issues for *k*NN ensembles, we introduce a novel multimodal perturbation approach, termed RRSB. The main idea is to simultaneously encourage diversity and individual accuracy within an ensemble classifier. There are four perturbations in RRSB. First, we perturb the learning parameter of distance metric. A weighted heterogeneous distance metric is proposed, which takes the importance of different attributes into consideration during the distance calculations. Second, we perturb the attribute space. The attribute reduction technique is used to reduce irrelevant attributes after the random subspace method. Third, we perturb the output target of *k* neighbors. The Dempster-Shafer theory is introduced to compute the output of the *k* neighbors' prediction. Finally, we perturb the training samples. The Bagging technique is used, which can improve the performance of the *k*NN ensemble when combined with the other perturbations.

The main contributions of this paper can be summarized as follows:

1) A weighted heterogeneous distance metric called WHDM is proposed;

2) The evidence theory and WHDM-based progressive *k*NN is used as a base classifier;

3) Random subspace and attribute reduction are used to perturb the attribute space; and

4) Combining several different perturbations, RRSB is proposed for *k*NN ensemble.

RRSB uses the idea of multimodal perturbation to generate diverse component classifiers, which has been previously published [2, 4, 15, 16, 17, 18]. However, RRSB considers the weight of attributes during the distance calculations and removes the irrelevant attributes for constructing each component classifier, which guarantees the accuracy of the component classifiers. Furthermore, RRSB can save on computational costs compared with GA- and PSO-based methods.

In the experimental stage, we first demonstrate the superior performance of evidence theory- and WHDM-based *k*NN compared with other types of *k*NN. We then analyze the influence of different perturbations and the effect of ensemble size on RRSB. Finally, we compare RRSB with state-of-the-art multimodal perturbation-based *k*NN ensemble methods on multiple UCI data sets and apply RRSB to network intrusion detection. The experimental results demonstrate the effectiveness of our algorithms.

The remainder of this paper is organized as follows. Section 2 presents neighborhood rough sets and an evidence theory-based *k*NN algorithm. Section 3 shows WHDM and an evidence theory-based weighted *k*NN algorithm. Section 4 presents the RRSB algorithm. Experiments are given in Section 5, and Section 6 concludes the work and raises several issues for future work.

2. Preliminaries

2.1 Neighborhood rough sets

The classical rough set theory proposed by Pawlak [19] has been proven to be an effective mathematical tool for dealing with uncertain and inaccurate data, especially for attribute selection. It employs a dependency function to evaluate the classification quality of a subset of attributes. However, this model is only applicable to nominal data. In practical problems, it is most often the case that the values of features may be both crisp and real-valued.

Neighborhood rough set theory is an extension of traditional rough set theory. The core idea of rough set theory is based on approximation and granules. To define the approximation of mixture data, a neighborhood relation can be used to generate a family of neighborhood granules characterized with numerical features. Yao [20] discussed the relationship between neighborhood operations and rough approximation operations and presented a neighborhood rough set model by using a distance function. The neighborhood rough set model was used for classification and attribute reduction by Hu et al. [21, 22], which could handle a knowledge classification system with not only continuous data but also with categorical data.

Next, we introduce some basic notions of neighborhood rough sets. Formally, the structure data used for the classification task can be written as a decision table, denoted by $DT = (U, C, D)$, where U is a nonempty finite set of instances $\{x_1, x_2, \ldots, x_n\}$, called a universe. In neighborhood rough set theory, the attribute set and class label are often put together for analyzing the inner structure and relation of samples, i.e., $A = C \cup D$, where C is the attribute set, and is a nonempty finite set of attributes $\{a_1, a_2, \ldots a_m\}$ to characterize the instance (sample), and D denotes class label ($D = \{c\}$). In other words, A is the union of predictor variables and class variable. For a given sample, each a_i has a determined value for characterizing the sample, and the class label c represents the class to which the sample belongs.

Definition 1. Given arbitrary $x_i \in U$ and $B \subseteq C$, the neighborhood $\delta_B(x_i)$ of x_i in the input space B

is defined as [22]: $\delta_B(x_i) = \{x_j \mid x_j \in U, \nabla_B(x_i, x_j) \le \delta\}$, where ∇ is a distance metric.

Definition 2. Let $B \subseteq A$ and $C \subseteq A$ be the numerical attributes and nominal attributes, respectively. The neighborhood granule of instance x induced by B, C and $B \cup C$ are defined as follows [22]:

1) $\delta_B(x) = \{x_i \mid \nabla_B(x, x_i) \le \delta, x_i \in U\}$;

2) $\delta_C(x) = \{x_i \mid \nabla_C(x, x_i) = 0, x_i \in U\}$;

3) $\delta_{B\cup C}(x) = \{x_i \mid \nabla_B(x, x_i) \le \delta \wedge \nabla_C(x, x_i) = 0, x_i \in U\}$, where $\wedge$ means "and" operator.

The first equation is used to handle numerical features, the second equation is based on classical rough sets, which is used to address nominal features, and the last equation can handle both numerical and nominal features, which is the most important part of neighborhood rough set theory.

Definition 3. Given a set of instances U and its neighborhood relation N over U, we call (U, N) a neighborhood approximate space. For any $X \subseteq U$, one subset of instance, called the lower approximation of X in (U, N), is defined as [22]:

$$\underline{N}X = \{x_i \mid \delta_B(x_i) \subseteq X, x_i \in U\}.$$

Clearly, $\underline{N}X \subseteq X$.

Definition 4. Given the neighborhood decision table $NDT = (U, C, D, N)$, $X_1, X_2, \ldots, X_N$ are the instance sets with decisions 1 to N, i.e., all instances in set X_j have the same decision label, and there are a total N different decision labels. $\delta_B(x_i)$ is the neighborhood information granule generated by attributes $B \subseteq C$. The lower approximation of decision D with respect to attributes B, also called the positive region of decision D with respect to B ($POS_B(D)$), is defined as [22]:

$$\underline{N_B}D = \cup_{j=1}^{N} \underline{N_B}X_j = POS_B(D)$$

where

$$\underline{N_B}X = \{x_i \mid \delta_B(x_i) \subseteq X, x_i \in U\}.$$

Definition 5. Given the neighborhood decision table $NDT = (U, C, D, N)$, distance function $\nabla$and neighborhood size δ, the dependency degree of D to B is defined as [22]:

$$\gamma_B(D) = |POS_B(D)|/|U|$$

where $|\bullet|$ is the cardinality of a set. $\gamma_B(D)$ reflects the ability of B to approximate D. As $POS_B(D) \subseteq U$, we have $0 \le \gamma_B(D) \le 1$.

Definition 6. Given a neighborhood decision table $NDT = (U, C, D, N)$, $B \subseteq C$, and $\forall a \in B$, we can define the significance of attribute a with respect to B and D as [22]:

$$SIG(a, B, D) = \gamma_B(D) - \gamma_{B-\{a\}}(D).$$

Table 1

Neighborhood decision table

Sample	a_1	a_2	a_3	D
x_1	5.8	2.7	y	Virg
x_2	6.3	3.3	n	Virg
x_3	7.1	3.0	n	Virg
x_4	6.9	3.1	y	Vers
x_5	7.0	3.2	y	Vers
x_6	6.4	3.2	n	Vers
x_7	6.1	2.8	y	Virg
x_8	6.0	3.0	y	Vers

Next, we give an example of computing the significance of attributes. Given a neighborhood decision table, shown as Table 1. The domain U contains eight samples, i.e., U = {x_1, …, x_8}.There are three attributes, including two continuous attributes (a_1, a_2) and one nominal attributes (a_3).

We first normalized two continuous attributes into interval [0, 1], then let B_1 = {a_1}, B_2 = {a_2}, and B_{12} = {a_1, a_2}, we obtained the neighborhood granules of each sample induced by B_1, B_2 and B_{12} (neighborhood radius δ = 0.2) by using Definitions 1 and 2(1). By introducing the nominal attribute subset B_3 = {a_3}, we obtained the partition U/B_3 = {{x_1, x_4, x_5 x_7, x_8}, {x_2, x_3, x_6}} by using Definitions 1 and 2(2). Let B_{13} = {a_1, a_3}, B_{23} = {a_2, a_3}, and A = {a_1, a_2, a_3}, then the neighborhood granules of each sample induced by B_{13}, B_{23}, and A were calculated by using Definition 2(3). The neighborhood granules of each sample induced by these attribute subsets are shown in Table 2.

Table 2

Neighborhood relation of samples

N	B_1	B_2	B_3	B_{12}	B_{13}	B_{23}	A
$\delta_{Bi}(x_1)$	{x_1, x_8}	{x_1, x_7}	{x_1, x_4, x_5, x_7, x_8}	{x_1}	{x_1, x_8}	{x_1, x_7}	{x_1}
$\delta_{Bi}(x_2)$	{x_2, x_6, x_7}	{x_2, x_5, x_6}	{x_2, x_3, x_6}	{x_2, x_6}	{x_2, x_6}	{x_2, x_6}	{x_2, x_6}
$\delta_{Bi}(x_3)$	{x_3, x_4, x_5}	{x_3, x_4, x_8}	{x_2, x_3, x_6}	{x_3, x_4}	{x_3}	{x_3}	{x_3}
$\delta_{Bi}(x_4)$	{x_3, x_4, x_5}	{x_3, x_4, x_5, x_6, x_8}	{x_1, x_4, x_5, x_7, x_8}	{x_3, x_4, x_5}	{x_4, x_5}	{x_4, x_5, x_8}	{x_4, x_5}
$\delta_{Bi}(x_5)$	{x_3, x_4, x_5}	{x_2, x_4, x_5, x_6}	{x_1, x_4, x_5, x_7, x_8}	{x_4, x_5}	{x_4, x_5}	{x_4, x_5}	{x_4, x_5}
$\delta_{Bi}(x_6)$	{x_2, x_6}	{x_2, x_4, x_5, x_6}	{x_2, x_3, x_6}	{x_2, x_6}	{x_2, x_6}	{x_2, x_6}	{x_2, x_6}
$\delta_{Bi}(x_7)$	{x_2, x_7, x_8}	{x_1, x_7}	{x_1, x_4, x_5, x_7, x_8}	{x_7}	{x_7, x_8}	{x_1, x_7}	{x_7}
$\delta_{Bi}(x_8)$	{x_1, x_7, x_8}	{x_3, x_4, x_8}	{x_1, x_4, x_5, x_7, x_8}	{x_8}	{x_1, x_7, x_8}	{x_4, x_8}	{x_8}

The domain U was divided into two parts by decision D, i.e., U/D = {{x_1, x_2, x_3, x_7}, {x_4, x_5, x_6, x_8}}. Let X_1 = {x_1, x_2, x_3, x_7}, X_2 ={x_4, x_5, x_6, x_8}, we computed the lower approximation and the positive region (POS) induced by attribute subsets B_{12} and A according to Definitions 3 and 4 through following procedures.

$\underline{N}_{B_{12}}X_1 = \{x_1, x_7\}$, $\underline{N}_{B_{12}}X_2 = \{x_5, x_8\}$;

$\underline{N}_{B_{12}}D = \underline{N}_{B_{12}}X_1 \cup \underline{N}_{B_{12}}X_2 = \{x_1, x_5, x_7, x_8\} = POS_{B_{12}}(D)$.

Next, we computed the positive region induced by attribute set A.

$\underline{N}_A X_1 = \{x_1, x_3, x_7\}$, $\underline{N}_A X_2 = \{x_4, x_5, x_8\}$;

$\underline{N}_A D = \underline{N}_A X_1 \cup \underline{N}_A X_2 = \{x_1, x_3, x_4, x_5, x_7, x_8\} = POS_A(D)$.

Since {a_3} = A – B_{12}, we computed the significance of the attribute a with respect to A and D by using Definitions 5 and 6.

$\gamma_A(D) = |POS_A(D)|/|U| = 6/8$, $\gamma_{A-\{a_3\}}(D) = \gamma_{B_{12}}(D) = \left|POS_{B_{12}}(D)\right|/|U| = 4/8$;

$SIG(a_3, A, D) = \gamma_A(D) - \gamma_{A-\{a_3\}}(D) = 6/8 - 4/8 = 0.250$.

By calculating the values of $\gamma_{A-\{a_1\}}(D)$ and $\gamma_{A-\{a_2\}}(D)$ using the similar procedure in previous, we obtained that $SIG(a_1, A, D) = 0$, $SIG(a_2, A, D) = 0.375$.

2.2 D-S theory-based *k*-nearest neighbor classifier

Evidential *k*NN (E*k*NN) classification proposed by Denoeux [23] is an improvement of traditional *k*NN. Compared with voting-based *k*NN, E*k*NN can obtain a smooth output result of the *k* neighbors' prediction. E*k*NN is based on the Dempster-Shafer (D-S) theory (also called evidence theory), which takes the distance value and class label together to determine the final class label. The main idea of E*k*NN is explained as follows. For a pending sample *t*, we seek *k* different neighbors of *t* in the training set at first, then we regard each neighbor as a piece of evidence in D-S theory, which is used as a support degree to support each class that *t* belongs to. Finally, we use the basic probability assignments from *k* neighbors to determine which class *t* belongs to.

Next, we explain the principle of E*k*NN. For a classification problem, let $\Omega = \{w_1, \ldots, w_c\}$ be the set of class labels (Ω is called *frame of discernment*), and $T =\{(x_1, L(x_1)), \ldots, (x_n, L(x_n))\}$ be the training samples, where for any $1 \le i \le n$, x_i and $L(x_i) \in \Omega$ are the training sample and class label, respectively. For a test sample t_s, let $F_s = \{(y_1, L(y_1)), \ldots, (y_k, L(y_k))\} \subseteq T$ be the set of the *k* neighbors of t_s in *T*, where for $1 \le i \le k$ each neighbor y_i of t_s has a class label $L(y_i)$. For any $1 \le i \le k$, if we suppose that $L(y_i) = w_q \in \Omega$, then (y_i, w_q) can be treated as an individual piece of evidence in favor of the classification of t_s, and we can use the following *basic probability assignment* (*BPA*) functions in Eqs. 1 and 2 to express the information contained in (y_i, w_q),

$$m^{s,i}(\{w_q\}) = \alpha, \tag{1}$$

$$m^{s,i}(\Omega) = 1 - \alpha. \tag{2}$$

where $\alpha \in [0, 1]$. The value of α is determined by the distance *d* between y_i and t_s (i.e., the increase of *d* results in the increase of α), and we can use a similarity function to describe the relation between α and *d*. In reference [23], Denoeux defined the similarity function as follows:

$$\alpha = \alpha_0 \cdot e^{-\gamma_q \cdot d^2}, \tag{3}$$

where α_0 and γ_q are two given parameters, $0 < \alpha_0 <1$ and $\gamma_q > 0$. For any $1 \le i, j \le k$ $(i \ne j)$, $m^{s,i}$ and $m^{s,j}$ are independent from each other, since they are induced by different training samples. By using the combination rule of Dempster described in Eq. 4, the orthogonal sum of *k* belief structures $m^{s,1}, \ldots, m^{s,k}$ can be combined together.

$$m^s = \bigoplus_{y_i \in F_s} m^{s,i}. \tag{4}$$

The combination of any two pieces of evidence in Eq. 4 can be written by Eq. 5.

$$(m_1 \oplus m_2)(C) = \frac{\sum_{A \cap B = C} m_1(A) m_2(B)}{1 - \sum_{A \cap B = \emptyset} m_1(A) m_2(B)}, \tag{5}$$

where $A \in \Omega$ and $B \in \Omega$, $m_1(A)$ and $m_2(B)$ are the basic probabilities. The basic probability of their conjunction $C = A \cap B$ is proportional to $m_1(A) \times m_2(B)$. The computation of Eq. 5 represents the sum over all conjunctions of arguments which support *C*. Finally, we can classify t_s through computing the belief degree function of each class label based on m^s. For instance, the joint decision may be the class w_q, which gets the maximum belief value formulated as Eq. 6,

$$w_q = \arg\max_{w_i} \sum_{F_j \subseteq \Omega} m(F_j). \tag{6}$$

For more detailed information about E*k*NN, please refer to [23].

3. Weighted heterogeneous distance metric for *k*NN algorithm

3.1 Weighted heterogeneous distance metric

*k*NN classification has no independent training stage, but when a pending sample is given to be classified, the algorithm will seek *k* nearest training samples through calculating distances and then use the majority voting to make a classification decision. Euclidean distance is often used to measure the distance between each pair of instances. Euclidean distance is a special case of Minkowski distance, which is defined in Eq. 7 [24], where x_1 and x_2 are two instances characterized by *d* dimensional attribute vectors. Let *p* = 2, it is Euclidean distance.

$$Minkowski_p(x_1, x_2) = (\sum_{n=1}^{d} |x_{1,n} - x_{2,n}|^p)^{1/p} \tag{7}$$

Minkowski distance is suitable for continuous or numerical attributes, but it cannot address nominal attributes. For nominal attributes, Stanfill and Waltz [25] proposed value difference metric (VDM), which works well in many nominal domains, but it does not handle continuous attributes directly. Instead, it uses a discretization method, which may lead to information loss and degrade the generalization performance. Many real world applications have both nominal and continuous attributes, for example, over half of the datasets in the UCI machine learning data repository. To address this issue, Wilson and Martinez [26] proposed HEOM and HVDM for heterogeneous attributes. HEOM and HVDM are defined as follows [26]:

$$HEOM(x, y) = \sqrt{\sum_{a=1}^{m} d_a(x, y)^2} \tag{8}$$

where $d_a(x, y)$ is defined as follows:

$$d_a(x, y) = \begin{cases} overlap_a(x, y) & \text{if } a \text{ is a nominal attribute,} \\ rn\_diff_a(x, y) & \text{if } a \text{ is a numerical attribute,} \\ \quad 1 & \text{if the value of } a \text{ on } x \text{ or } y \text{ is unknown} \end{cases} \tag{9}$$

where $overlap_a(x, y) = \begin{cases} 1 & \text{if } N(x, a) \neq N(y, a), \\ 0 & \text{otherwise.} \end{cases}$ and $\text{rn}\_diff_a(x, y) = \frac{|x-y|}{range_a}$.

$$HVDM(x, y) = \sqrt{\sum_{a=1}^{m} d_a(x, y)^2} \tag{10}$$

where $d_a(x, y)$ is defined as follows:

$$d_a(x, y) = \begin{cases} normalized\_vdm_a(x, y) & \text{if } a \text{ is a nominal attribute,} \\ normalized\_\ diff_a(x, y) & \text{if } a \text{ is a numerical attribute,} \\ \quad 1 & \text{if the value of } a \text{ on } x \text{ or } y \text{ is unknown} \end{cases} \tag{11}$$

where $vdm_a(x, y) = \sum_{c=1}^{C} |P_{a,x,c} - P_{a,y,c}|^2$, $P_{a,x,c}$ is the conditional probability that the output class is *c* given that attribute *a* has the value *x*.

The biggest difference between HEOM and HVDM is that HEOM uses the overlap metric for nominal attributes but HVDM adopts the VDM for nominal attributes.

In some situations, the effect of some attributes may be higher than other attributes [27]. For example, in a tourism recommendation system, a sample (*person*) has seven conditional attributes (*gender*, *age*, *salary*, *house*, *number of children*, *job*, and *education level*) in the information system. The attributes of *salary* and *number of children* may have more weight than the other five attributes in determining whether to travel, and the attribute of *education level* may have the lowest importance in the decision. Therefore, an attribute-weighted measure scheme needs to be designed for distance metrics, which can take the importance of different attributes into consideration.

The above idea of weighting attributes in distance metric has been successfully applied to metric learning. For instance, an efficient multi-modal geometric mean metric learning (EMGMML) structure to deal data with multiple modalities was proposed by Liang et al. [42]. In EMGMML, each modality is assigned a weight to emphasize the difference of multi-modalities. Zhai et al. [43] proposed parametric local multi-view hamming distance metric learning (PLMH) based on a set of local hash functions, in which different local hash functions are learned at different positions in the input feature space. A novel distance metric learning method by fusing multiple features was investigated by Lv and Duan [44], which can learn the distance metric on single feature as well as the weights of different features in a joint framework. By maximizing the Jeffrey divergence between two multivariate Gaussian distributions for linear transformations, an optimization framework for distance metric learning was proposed by Nguyen et al. [45]. Wang et al. [46] proposed a new weakly supervised distance metric learning method, called multi-view metric learning (MML), which integrates compatible and complementary information from multiple views using KL-divergence. Most of these methods used multi-view for metric learning, our method weights each feature directly by using the attribute significance in rough sets, which can mine the intrinsic information.

Neighborhood rough sets proposed by Hu et al. [22], as an extension of classical rough sets, has been widely used in pattern analysis and feature metrics. In neighborhood rough set, the significance of attribute *a* is defined as definition 6, that is, the dependency degree of all attributes *B* to class label minus the dependency degree of attributes $B - \{a\}$ to class label. In other words, the attribute significance of *a* represents the importance of the classification task; the bigger the attribute significance of *a*, the more important the attribute *a* is. Therefore, we can use the attribute significance to design a weight scheme in distance metrics. Based on the above, a weighted heterogeneous distance metric (WHDM) is proposed in definition 7.

Definition 7. (Weighted Heterogeneous Distance Metric) Given a neighborhood decision table *NDT* = (*U*, *C*, *D*, *N*), for any two samples $x, y \in U$, the distance *whd* between *x* and *y* is defined as follows:

$$whd(x, y) = \sum_{a \in C} weight(a) \times d_a(x, y) \tag{12}$$

where *weight*(*a*) reflects the weight of attribute *a*, $d_a(x, y)$ represents the distance between *x* and *y* with respect to attribute *a*, and $d_a(x, y)$ and *weight*(*a*) are defined as follows:

$$d_a(x, y) = \begin{cases} overlap_a(x, y) & \text{if } a \text{ is a nominal attribute,} \\ diff_a(x, y) & \text{if } a \text{ is a numerical attribute,} \\ 1 & \text{if the value of } \ a \text{ on } x \text{ or } y \text{ is unknown} \end{cases} \tag{13}$$

where $overlap_a(x, y) = \begin{cases} 1 & \text{if } N(x, a) \neq N(y, a), \\ 0 & \text{otherwise.} \end{cases}$ and $diff_a(x, y) = \frac{|N(x,a) - N(y,a)|}{4\gamma_a}$. We define the weight of attribute *a* as follows:

$$weight(a) = \frac{SIG(a)}{max_{a \in C} SIG(a)} \tag{14}$$

In the above definition, the distance $d_a(x, y)$ can be used for both nominal and numerical attributes, where the overlap metric is used for nominal attributes, the normalized Manhattan metric is used for numerical attributes, and when the value of attribute *a* on *x* or *y* is unknown, the value of $d_a(x, y)$ is assigned as 1. In $diff_a(x, y)$, γ_a is the standard deviation of the numeric values of attribute *a*; this can avoid the outlier influencing the distance by the range of that attribute, where *N*(*x*, *a*) and *N*(*y*, *a*) represent the values of *x* and *y* on attribute *a*, respectively.

In Eq. 14, the *SIG*(*a*) denotes the significance of attribute *a* in a neighborhood decision system, and the *weight*(*a*) used to characterize the weight of attribute *a* in WHDM is calculated based on *SIG*(*a*). It is obvious that if *SIG*(*a*) > 0 then *weight*(*a*) is proportional to *SIG*(*a*), and *weight*(*a*) ranges from 0 to 1.

3.2 The WHDM-based *k*NN algorithm and its extension to D-S theory

In this section, based on the proposed WHDM in the previous section, we use the WHDM as a distance metric for *k*NN. It can address heterogeneous attributes and consider the importance of attributes in calculating the distance. Different from the traditional *k*NN, weighted *k*NN adequately takes the weight of different attributes into account, and weighted *k*NN can work well with both continuous attributes and nominal attributes. The detailed procedure of weighted *k*NN is described in Algorithm 1.

Algorithm1: The weighted heterogeneous distance metric-based *k*NN
Input: neighborhood decision table $NDT = (U, C, D, N)$, where $|U| = n$, $|C| = m$; the number of neighbors k; the instance x to be classified; δ (controls the size of the neighborhood).
Output: the class label of instance x.

```
For any attribute $a \in C$ do:
    Compute the positive region $POS_C(D)$ and $POS_{C-\{a\}}(D)$ by using Definition 4
    Compute the dependency degree $\gamma_C(D)$ and $\gamma_{C-\{a\}}(D)$ by using Definition 5
    Compute the significance of attribute $a$ $SIG(a) = \gamma_C(D) - \gamma_{C-\{a\}}(D)$ by using Definition 5
End for
Find $SIG(a_k)$ such that $SIG(a_k) = \max_i SIG(a_i)$, and denote it as $MAX$
For each attribute $a \in C$, compute the weight of $a$ $weight(a) = SIG(a)/MAX$
For any $y_i \in U$ $(1 \le i < k+1)$ do:
    For any attribute $a \in C$ do:
        Compute the distance $d_a(x, y_i)$ between $x$ and $y_i$ on attribute $a$
    End for
    Compute the distance $whd(x, y_i) = \sum_{a \in C} weight(a) \times d_a(x, y_i)$ between $x$ and $y_i$ on $C$
    Deposit $i$ and $whd(x, y_i)$ into map $D$. //$D$ is a map structure, it contains two value (key and
      vlaue), i.e., each key maps to a value, where $D_{[i]}.key = i$, and $D_{[i]}.value = whd(x, y_i)$,
      respectively.
    Sort map $D$ by $D.value$ in ascending order. //$D_{[k]}.value$ is largest.
End for
For any $y_i \in U$ $(k+1 \le i < n)$ do
    Compute the distance $whd(x, y_i)$ between $x$ and $y_i$
    If $whd(x, y_i) < D_{[k]}.value$, then let $D_{[k]}.key = i$, $D_{[k]}.value = whd(x, y_i)$, and rearrange map $D$
      by $D.value$ in ascending order. //The aim is to make $D_{[k]}$ be the farthest in the neighbors of
      $x$.
End for
Count the number of class label of instances in map $D$
Find the class label $c$, which has the maximum number
Return the predicted class label $c$
```

As described in Section 2.2, Dempster-Shafer theory-based E*k*NN can provide a soft combination result of *k* different neighbors [23]. To further explore the development of weighted *k*NN, an evidence theory-based weighted *k*NN algorithm is developed to improve the output combination of *k* neighbors' prediction. Compared with weighted *k*NN, evidence theory-based weighted *k*NN only modifies the output target of *k* neighbors by using Dempster-Shafer theory; the combination of *k* neighbors' prediction in evidence theory-based weighted *k*NN is the same as that in E*k*NN. Considering that E*k*NN has been used in many studies [17, 18, 23, 28] and that Section 2.2 describes the principles of E*k*NN, here we give a brief description of evidence theory-based weighted *k*NN as follows:

1) Compute the distance based on WHDM;

2) Treat the class information of *k* neighbors and distances as evidence. Use the *BPA* functions in

Eqs. 1 and 2 to generate evidence rules;

3) Fuse these evidence rules using Dempster rules in Eqs. 4 and 5;

4) Compute the belief degree of possible class labels according to the belief function in Eq. 6; and

5) Make a decision based on Eq. 6.

4. RRSB

Bagging used for *k*NN ensemble can hardly increase the generalization performance, which has been proven by Breiman [9]. This is because Bagging uses a bootstrap sampling technique to generate accurate but diverse component classifiers, which is effective on unstable classifiers such as decision trees and neural networks.

To address this issue, several different methods have been proposed. For instance, Zhou and Yu [2] used bootstrap sampling, attribute filtering and randomly configured distance metrics for *k*NN ensembles. Nanni and Lumini [18] proposed a PSO-based multimodal perturbation for *k*NN ensembles, where RSM is used to perturb the attribute space, PSO is adopted to perturb the learning parameters of each component classifier, and evidence theory is used for combining the outputs of *k* different neighbors. Altinçay [17] proposed a GA-based multimodal perturbation for *k*NN ensembles, which uses GA to jointly estimate both the best-fitting attribute subsets and learning parameters of each member classifier.

RSM used in [2] and [18] can increase the diversity of component classifiers, but it might damage the accuracy. GA- and PSO-based multimodal perturbation methods [17, 18] cost too much time, and these methods do not perturb the training data. In this paper, we propose a novel multimodal perturbation method named RRSB for *k*NN ensembles. In detail, we use a weighted heterogeneous distance metric to perturb the learning parameter, adopt the attribute reduction technique to reduce irrelevant attributes after the random subspace method, which perturbs the attribute space, and introduce the Dempster-Shafer theory into the output target of *k*NN, which increases the performance of the component classifiers. With the above three perturbations, we adopt Bagging to perturb the training data. The main contribution of RRSB is that the multimodal perturbation-based RRSB not only increases the diversity of member classifiers but also guarantees the accuracy of member classifiers.

RSM, as first proposed by Ho [3], was used for decision tree ensembles. Ho [4] further applied RSM to *k*NN ensembles. For *k*NN classification, when calculating distances between a pending sample and training samples, only attributes corresponding to the selected subspace are used. In other words, each component classifier in the ensemble model corresponds to a random subspace, and in the ensemble stage, the predicted results of different classifiers are combined by majority voting. Ho [4] explained why RSM can work on *k*NN as follows: RSM is a derivative of *stochastic discrimination* where many stochastically generated weak member classifiers are combined to obtain nearly monotonic increase in accuracy [29]. The member classifiers do not have full discriminative power but they generalize very well to unseen data for the same problem. A stochastic procedure is adopted to introduce independence among the member classifiers. Combining their decisions together leads to increased discriminative power. RSM follows the same approach. By ignoring some dimensions of the attribute space, invariance of classification is maintained for samples that ignore different dimensions. By randomly selecting the combination of dimensions to be ignored, certain independence is introduced among the component classifiers. By combining the component decisions, discriminative power of ensemble classifier is improved.

RSM improves the performance of *k*NN ensembles through increasing the diversity of component classifiers by injecting randomness. However, the accuracy of the component classifiers trained on the random subspace data is not sufficient, because redundancy may exist in attributes generated by RSM, and the redundant attributes that are irrelevant to the learning target may disturb the learning on the relevant attributes, which is harmful for the accuracy of component classifiers. The component classifiers are trained on subsets generated by RSM, and the irrelevant attributes might influence the accuracy of component classifiers. To address this issue, we use the attribute reduction method in neighborhood rough sets to remove irrelevant attributes after employing RSM. Here, we remove irrelevant attributes with the attribute reduction method after employing RSM rather than before, because all of the remainding attributes after attribute reduction might be indispensable, and employing RSM on reduced attribute space will not guarantee the accuracy of component classifiers. In this paper, we use F2HARNRS [22] to reduce irrelevant attributes from the attribute space generated by RSM. The procedure of the F2HARNRS algorithm is shown as Algorithm 2.

Algorithm 2: F2HARNRS

Input: Data set $T = (U, C, D, N)$, where $|U| = n$, $|\mathrm{C}| = m$; δ (controls the size of the neighborhood).

Output: Reduct *red*.

Initialization: $red \leftarrow \phi$, $S \leftarrow U$, where *red* denotes the reduct of *C*, and *S* denotes the set of samples out of positive region *POS*, where *POS* as defined in Definition 4 is used to compute reduct *red*.

```
While $S \neq \phi$ do:
    For each $a_i \in C - red$ do:
        Generate a temporary decision table $T_i = (U, red \cup a_i, D, N)$
        $POS_i \leftarrow \phi$
        For each sample $O_j \in S$ do
            Compute $\delta(O_j)$ in the neighborhood decision table $T_i$
            If there exists $X_k \in U/D$ such that $\delta(O_j) \subseteq X_k$
                $POS_i = POS_i \cup O_j$
            End if
        End for
    End for
    Find $a_k$ such that $|POS_k| = max_i |POS_i|$
    If $POS_k \neq \phi$
        $red = red \cup a_k$
        $S = S - POS_k$
    Else
        Exit while
    End if
End while
Return red
```

Based on the above, a multimodal perturbation-based ensemble learning method named RRSB is proposed, where evidence-based weighted *k*NN is used as a base classifier. In detail, RRSB includes the following steps:

1) Given a training set *T* with *C* dimensional attributes, we randomly select attributes from *C* to

form a subset C_{sub};

2) We use the attribute reduction method (F2HARNRS) to reduce the irrelevant attributes in C_{sub}, and hence a reduced attribute subset $C_{sub\text{-}red}$ of C_{sub} is generated;

3) Resampling samples from T via bootstrap sampling to generate a new sample set T_{smp};

4) A member classifier C is obtained through training T_{smp} on the attribute subset $C_{sub\text{-}red}$ using the evidence theory-based weighted *k*NN;

5) Repeat the above steps t times to obtain t member classifiers; and

6) The obtained t member classifiers are combined into an ensemble by majority voting.

The detailed procedure of RRSB is described in Algorithm 3.

Algorithm 3: RRSB

Input: Training set $T = (U, C, D, N)$, where $|U| = n$, $|C| = m$; the number of member classifiers t, the ratio of random subspace r.

Output: Ensemble classifier EC.

Initialization: $BC \leftarrow \phi$, $T_{smp} \leftarrow \phi$, where BC denotes the set of all member classifiers, and T_{smp} denotes the temporary training set;

```
For $i = 1$ to $t$ do:
    Randomly select attributes from $C$ to form attribute subset $C_{sub}$ such that the ratio of $C_{sub}$ to $C$ is $r$
    Use the algorithm 2 to reduce $C_{sub}$, to get a reduced set $C_{sub\text{-}red}$ of $C_{sub}$
    $T_{smp} \leftarrow \phi$
    For $j = 1$ to $n$ do:
        Randomly select a sample $smp$ from $U$
        $T_{smp} = T_{smp} \cup smp$
    End for
    Construct a classifier $C$ by training on $T_{smp}$ corresponding to $C_{sub\text{-}red}$ using the given classification algorithm
    $BC = BC \cup C$
End for
Obtain an ensemble classifier $EC$ from the set $BC$ by voting
Return $EC$
```

In algorithm RRSB, evidence theory-based weighted *k*NN is used to train component classifiers, which can allow RRSB perturb the training set through multiple perspectives, i.e., perturbing the training data, attribute space, and learning parameter and output targets. We know that the complexity of *k*NN is $O(n \times m)$, where n is the number of instances in the training set, and m is the dimension of attribute space. Luo et al. [30] demonstrated that *k*NN is very applicable to the number of instances far larger than the dimension. In other words, the dimensions have a greater effect on the complexity for *k*NN. Calvo-Zaragoza et al. [31] reduced the training set by using a prototype selection method to decrease the computational cost. Prasartvit et al. [32] used the artificial bee colony method for dimension reduction and then used *k*NN for analysis, which can save much time. García-Pedrajas and Ortiz-Boyer [33] proposed multiple input space projections for Boosting *k*NN. The input space projection can save much time in calculating the distance, but Boosting, as an iterative algorithm, is time consuming. In RRSB, the attribute space is reduced by RSM and attribute techniques, which can decrease the time cost during distance computing. The

Bagging used for perturbing the training data can easily be implemented in parallel, which guarantees the efficiency of RRSB. The computational time of attribute reduction in Algorithm 2 is $O(m\times n\times \log n(k+1)/2)$ [22], where n is the number of samples in decision table T, m is the number of raw attributes in T and k is the number of attributes in a reduced attribute space. The computational cost of E*k*NN is linearly related to the number of classes [23], and evidence theory-based weighted *k*NN only weights the attributes in distance metric; therefore, the computational cost of that is c times as much as E*k*NN, that is $O(c\times n)$, where c is number of class labels. Finally, the complexity of RRSB is $O(t\times((c+m \log n(k+1)/2)\times n))$, where t is the ensemble size. In the experiment, we compared the testing time of RRSB with other methods.

5. Experimental results

5.1 Individual classifier performance

In this section, we compared weighted *k*NN (*k*NN$_1$) and evidence theory based weighted *k*NN (*k*NN$_2$) with other types of *k*NN, including traditional *k*NN, E*k*NN (*k*NN$_3$), HVDM-based *k*NN (*k*NN$_4$) and HEOM based-*k*NN (*k*NN$_5$). To verify the effectiveness of *k*NN$_1$ and *k*NN$_2$, multiple data sets from the UCI machine learning data repository [34] were used in the experiments. The size of data sets ranged from 208 to 11,500, and the number of attributes varied from 6 to 178. Details of the data sets are shown in Table 3.

Table 3

Summary of the data sets

No.	Data sets	Size	Attribute		Class
			categorical	continuous	
1	sonar	208	0	60	2
2	ionosphere	351	2	32	2
3	liver	345	0	6	2
4	vowel	990	3	10	11
5	vehicle	846	0	18	4
6	heart	303	8	5	5
7	wdbc	569	0	30	2
8	pima	768	1	7	2
9	credit-g	1000	21	3	2
10	cardiotocography	2126	14	26	10
11	thoracic	470	13	3	2
12	diabetic	1151	3	16	2
13	epileptic	11500	0	178	5
14	firm teacher	10800	16	0	4
15	pubchem	4279	114	30	2
16	biodegradation	1055	8	33	2
17	seismic-bumps	2584	12	6	2
18	turkiye student	5820	32	0	5
19	z-alizadeh	303	34	21	2
20	movement-libras	360	0	90	15

Table 4

Classification accuracy obtained using different *k*NN algorithms with *k* = 3, 5, 7 (in %)

Data	k = 3						*k* = 5						*k* = 7					
set	*k*NN	*k*NN$_5$	*k*NN$_4$	*k*NN$_1$	*k*NN$_3$	*k*NN$_2$	*k*NN	*k*NN$_5$	*k*NN$_4$	*k*NN$_1$	*k*NN$_3$	*k*NN$_2$	*k*NN	*k*NN$_5$	*k*NN$_4$	*k*NN$_1$	*k*NN$_3$	*k*NN$_2$
1	76.14	76.14	76.25	76.38	76.50	76.79	74.04	74.17	75.28	75.54	74.56	76.88	73.56	73.54	73.17	74.69	70.87	74.17
2	84.05	84.15	83.74	86.35	88.00	88.91	84.61	84.66	84.77	85.37	86.80	87.12	84.92	84.88	84.97	85.18	86.80	87.22
3	55.07	54.25	54.75	58.26	63.07	64.47	56.52	57.71	56.94	59.71	64.88	65.56	56.82	55.11	55.34	59.34	65.12	66.64
4	67.73	67.99	66.48	65.59	64.60	64.68	67.26	68.14	67.95	66.18	62.75	65.34	67.49	66.39	66.20	68.24	61.66	66.51
5	85.35	85.17	84.72	85.92	90.73	90.74	78.48	79.91	77.26	83.54	88.08	88.96	69.49	68.27	67.91	75.59	85.94	86.24
6	53.14	54.47	54.86	57.81	62.98	62.87	54.79	57.88	59.63	59.86.	64.30	65.57	53.83	57.74	59.84	59.71	63.84	64.73
7	97.01	96.63	96.32	96.87	92.57	95.14	97.01	96.58	96.97	96.59	92.82	95.94	96.49	95.39	95.26	95.24	93.17	94.35
8	70.72	70.96	71.09	72.66	69.95	71.58	72.92	73.38	73.97	73.14	72.37	73.37	72.01	73.08	72.96	73.33	73.20	74.06
9	71.73	72.18	71.17	73.35	67.98	72.43	71.42	72.54	72.89	72.16	69.19	72.84	72.61	73.18	73.29	72.95	69.82	71.17
10	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
11	82.55	82.98	82.76	81.97	82.34	82.46	83.83	83.88	83.70	82.95	83.14	83.36	83.75	83.17	82.54	81.76	82.59	83.25
12	60.97	61.07	60.83	60.86	61.14	62.96	61.68	61.87	61.36	60.95	62.24	62.73	62.43	62.75	61.87	61.06	63.15	63.96
13	48.23	47.16	47.26	49.97	49.26	51.54	45.33	44.37	44.25	47.91	46.87	48.86	43.97	42.17	42.33	44.56	44.10	46.68
14	78.87	79.27	80.16	78.94	79.96	80.68	79.48	80.16	80.87	80.16	80.95	81.45	79.57	78.26	78.15	80.73	80.09	80.89
15	98.51	98.57	98.64	98.81	98.42	98.36	98.58	98.17	98.29	98.11	98.17	98.39	98.63	98.70	98.68	98.69	98.54	98.36
16	83.72	84.51	84.46	83.97	83.79	84.86	84.67	85.13	85.27	84.89	85.02	86.14	84.71	85.11	85.28	84.99	84.85	85.83
17	92.40	93.17	93.38	92.94	93.16	93.18	93.10	93.76	93.58	93.71	93.89	93.14	93.33	93.59	93.97	92.87	92.88	93.06
18	82.23	83.85	82.27	81.54	82.96	83.03	82.80	83.27	82.17	81.13	82.28	81.97	82.75	82.99	81.25	81.11	82.85	82.07
19	79.20	79.87	80.11	79.94	79.56	80.57	81.32	81.72	81.94	81.35	81.75	81.96	81.99	82.16	82.35	82.27	82.76	82.96
20	61.11	59.36	58.42	62.08	63.17	61.96	59.44	58.87	58.11	60.72	60.71	58.94	53.22	51.16	52.34	53.36	51.87	52.08
Ave	76.44	76.59	76.38	77.21	77.51	78.36	76.36	76.81	76.76	78.11	77.54	78.43	75.58	75.38	75.39	76.28	76.71	77.71

There are two popular methods for calculating the generalization performance of an algorithm, i.e., hold-out and *K*-fold cross-validation techniques [35]. In the hold-out method, the data set is randomly separated into two parts, a training set and a test set. For the *K*-fold cross-validation method, the data set is randomly separated into *K* equal sized-parts, where the training process is carried out using *K*-1 parts, and the remaining part is used for computing the generalization performance. In general, the *K*-fold cross-validation usually needs to be performed several times to obtain mean results.

The *K*-fold cross-validation costs much time and the large number of instances used for training may result in small difference on classification results. In this paper, in order to be consistent with the processes of *k*NN and E*k*NN in the literature [17], we use the hold-out method in the experiments. In particular, for each data set, we repeated the experiments 50 times. Each time the data set was randomly separated into an equal-sized training set and test set, and the average of 50 results was computed. The Euclidean distance metric is used for the traditional *k*NN and *k*NN$_3$ algorithms, the WHDM is used for the *k*NN$_1$ and *k*NN$_2$ algorithms, and *k*NN$_4$ and *k*NN$_5$ use two different heterogeneous distance metrics.

Considering that the value of *k* can influence the classification accuracy, we tested the experiments with several *k* values (k = 3, 5, 7). For the D-S theory-based *k*NN$_2$ and *k*NN$_3$ algorithms, the values of parameter α_0 and γ's should be set, which is mentioned in Section 2.2. In our experiments, α_0 = 0.95 and γ's were equal to the inverse of mean distance among the training instances of the corresponding class, which were also used in [17]. Since we used the neighborhood rough set to compute the weighted heterogeneous distance in *k*NN$_1$ and *k*NN$_2$, we needed to set the size of neighborhood δ in the experiments, and the parameter δ for controlling the size of neighborhoods is set as 0.2, which has been proven to be a suitable value [22]. The classification accuracy of different types of *k*NN algorithms with various *k* values are listed in Table 4.

From Table 4, we can find the following results. First, when *k* was equal to 3, 5 and 7, *k*NN$_2$ obtained 9, 10 and 10 times the best individual accuracy, respectively, and *k*NN$_2$ always had the best accuracy on 8 data sets (No. 2, 3, 5, 12, 13, 14, 16 and 19) regardless of whether *k* was equal to 3, 5 or 7. Second, *k*NN$_2$ achieved the best average accuracy on three different *k* values compared with other types of *k*NN. Particularly, when the value of *k* was selected as 5, *k*NN$_2$ achieved the highest average accuracy. Third, being the same as *k*NN$_3$, *k*NN$_2$ is also not sensitive to the value of *k*; the maximum difference of mean accuracy in *k*NN$_2$ is 0.72%, i.e., the highest average accuracy (78.43%) minus the lowest average accuracy (77.71%). Finally, compared with *k*NN, *k*NN$_4$ and *k*NN$_5$, *k*NN$_1$ clearly improves the classification accuracy.

5.2 The influence of different perturbations on the performance of ensemble classifier

RRSB makes use of multimodal perturbations to generate accurate but diverse component classifiers. If we breakdown the RRSB algorithm, several variants of RRSB can be derived that can be used to analyze the influence of different perturbations on the performance. We first employ two types of perturbation, and BagE (Bagging + E*k*NN), BagW (Bagging + weighted *k*NN) and BagR (Bagging + RSM) are generated. Next, three types of perturbation are employed, and BagEW (Bagging + evidence theory + weighted *k*NN) and REW (RSM + evidence theory + weighted *k*NN) are generated. Finally, we employ four types of perturbation, RAEW (RSM + attribute reduction + evidence theory + weighted *k*NN) and RBEW (RSM + Bagging + evidence theory + weighted *k*NN) are obtained. These algorithms are summarized in Table 5.

Table 5

RRSB and its degraded variants

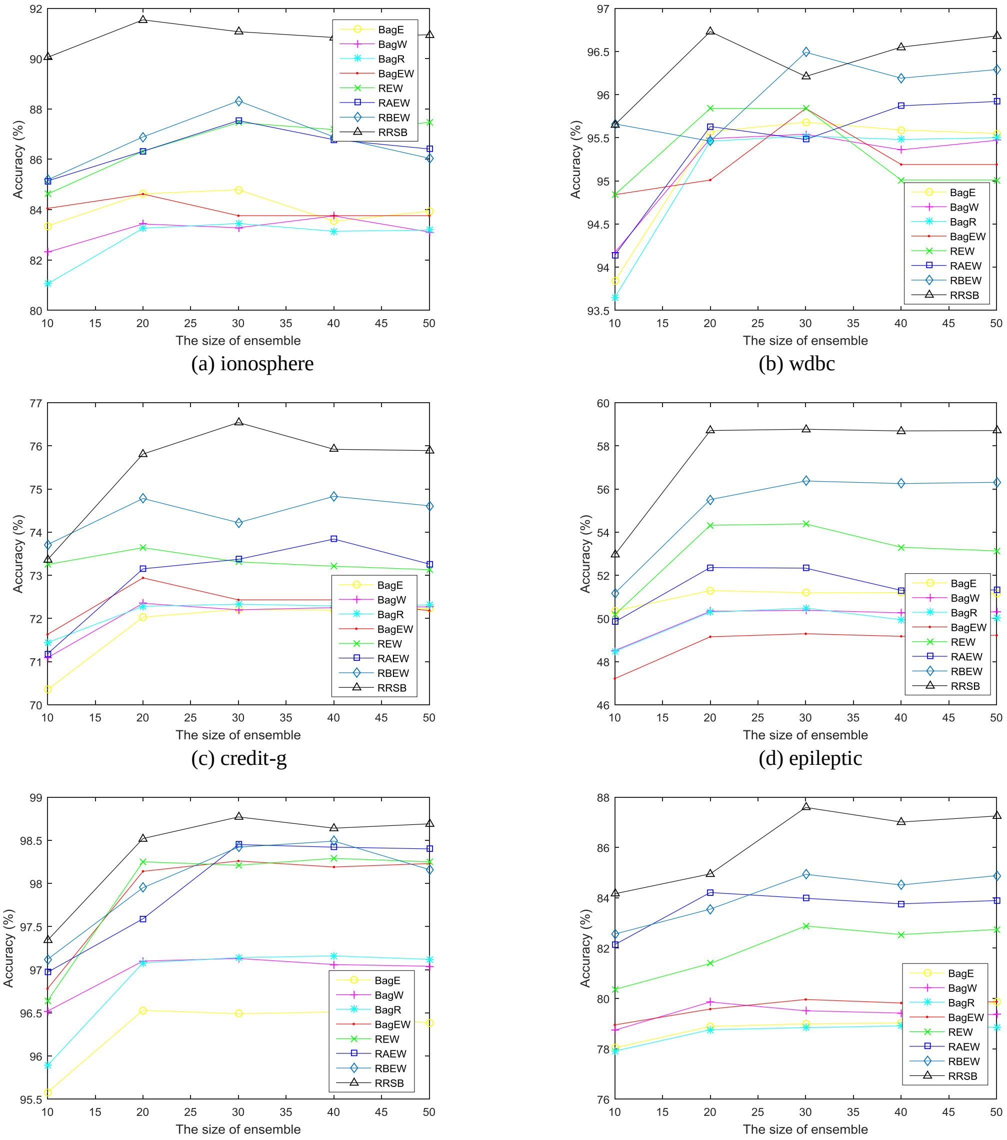
Methods	Perturb training data with	Perturb learning parameter with	Perturb attribute space with		Perturb individual output target with
	bootstrap sampling	WHDM	random subspace	attribute reduction	D-S theory
BagE	YES	NO	NO	NO	YES
BagW	YES	YES	NO	NO	NO
BagR	YES	NO	YES	NO	NO
BagEW	YES	YES	NO	NO	YES
REW	NO	YES	YES	NO	YES
RAEW	NO	YES	YES	YES	YES
RBEW	YES	YES	YES	NO	YES
RRSB	YES	YES	YES	YES	YES

Table 6

Classification accuracy of RRSB and its degenerated variants with k = 5 (in %)

Data sets	BagE	BagW	BagR	BagEW	REW	RAEW	RBEW	RRSB
sonar	75.97	76.51	75.15	76.44	76.92	78.84	78.37	83.98
ionosphere	84.62	83.43	83.26	84.62	86.32	86.32	86.89	91.54
liver	55.17	54.87	54.57	55.94	63.77	64.47	65.22	65.22
vehicle	68.14	69.25	68.78	68.91	70.75	71.86	70.75	73.51
vowel	81.25	83.56	83.15	84.14	88.82	90.35	90.72	91.29
heart	76.21	74.86	75.26	75.12	75.09	76.18	78.77	81.71
wdbc	95.64	95.58	95.58	97.01	96.84	96.73	95.26	96.73
pima	73.32	74.11	74.24	73.88	75.57	75.57	77.66	77.66
credit-g	72.18	72.20	72.21	72.94	73.64	73.15	73.78	75.81
cardiotocography	100.00	100.00	100	100.00	100.00	100.00	100.00	100.00
thoracic	83.51	84.45	83.14	84.58	86.27	87.38	88.17	91.37
diabetic	63.68	64.98	64.05	64.47	68.42	71.36	74.17	76.58
epileptic	51.25	50.36	50.25	49.25	54.32	52.38	55.55	58.75
firm teacher	79.85	79.56	79.56	80.18	79.65	81.87	81.14	83.36
pubchem	96.53	97.11	97.12	98.25	98.23	98.45	98.41	98.76
biodegradation	84.12	83.65	83.14	85.51	86.14	89.35	91.57	92.17
seismic-bumps	92.26	93.31	93.26	93.42	94.57	93.45	93.26	93.82
turkiye student	80.84	81.74	81.17	82.36	84.97	85.51	86.18	88.56
z-alizadeh	78.96	79.83	78.83	79.94	82.87	84.14	84.92	87.51
movement-libras	65.28	64.52	64.28	64.93	69.70	75.14	77.89	83.19
Ave	77.94	78.19	77.85	78.59	80.64	81.63	82.43	84.58

In this section, we compared RRSB with its variants. In the experiments, all parameters are the same as those in the previous experiments in Section 5.1. To simplify the experiments, we only did the experiments with k = 5 because of the best mean accuracy obtained in our previous experiments. The size of the ensemble is set as 25, which was found to be a reasonable value after exhaustive experiments [36]. Since the random subspace method needs to set the subspace ratio, here the subspace ratio r is randomly selected in the range [M/3, 2M/3] in each trial, where M denotes the number of attributes in each data set. Table 6 shows the classification accuracy of RRSB and its degenerated variants.



(a) ionosphere

(b) wdbc

(c) credit-g

(d) epileptic

(e) pubchem

(f) z-alizadeh

Fig. 1. Classification accuracies of different algorithms versus the size of ensemble on 6 data sets.

In contrast to the experimental results from Table 6, it can be seen that BagE and RRSB obtained the worst and the best average accuracy, respectively. The classification accuracy increased with the increase of perturbation, that is, the more perturbations are used, the greater is the diversity of the obtained component classifiers. Compared with the single kNN$_2$ in Table 4, the experimental results demonstrate that the single perturbation Bagging (BagE, BagW and BagR) can barely increase the classification accuracy, which has also been proven by Breiman [9]. However, when combined with other types of perturbations, the classification performance will obviously increase, that is, when compared with BagE, BagW and BagR, REW and RAEW obtained higher classification accuracies because kNN$_2$ is sensitive to the input space. Furthermore, RAEW achieved a higher accuracy than REW, since RAEW removed the irrelevant attributes that can influence the classification performance. Moreover, the perturbation of the attribute space and the training data-based RBEW and RRSB obtained higher classification accuracies compared with other algorithms. RRSB has the highest accuracy since the most perturbations are used on it to guarantee the diversity of member classifiers.

5.3 The impact of ensemble size on the performance of RRSB.

In the previous Section 5.2, for all ensemble algorithms, the ensemble size was set to 25, according to [32]. In this section, in order to further explore the influence of the ensemble size on the classification performance in the experiments, we did the experiments on different ensemble sizes. The ensemble size was set in the range [10, 50], with steps of 10. The other parameters were set in Section 5.2, and the experimental results are shown in Fig. 1.

As shown in Fig. 1, for all data sets, the classification accuracy of most algorithms initially increased, achieved at peak value, and then either decreased or remained stable. The size of ensemble with the best accuracies was 20 or 30, which was also demonstrated by Maclin and Opitz [36]. Compared with the other algorithms, RRSB achieved the best accuracies, and the accuracies of RRSB showed the least change when the ensemble size was increased, in other words, the size of ensemble had little influence on the classification performance.

5.4 Comparison with other multimodal perturbation based ensemble algorithms

In this section, we compared RRSB with other multimodal *k*NN ensemble algorithms, including FASBIR [2], GA [17] and EPSO [18]. These are all multimodal perturbation-based ensemble learning methods. Brief descriptions of these methods are given as follows:

1) FASBIR: This is a multimodal perturbation method, i.e., disturbing the learning parameter by randomly selecting a value of *p* in Minkowski distance function for each classifier, the training samples by Bagging and input attributes by random subspace method on filtered attributes [2].

2) GA: This is a multimodal perturbation method that uses GA to jointly estimate both the best-fitting attribute subsets and the learning parameters of each *k*NN classifier [17].

3) EPSO: This is a multimodal perturbation method, where RSM is used to perturb the attribute space, and PSO is adopted to perturb the learning parameters and the attribute subset of each base classifier [18].

4) RRSB: This is the proposed method in this paper. It is also a multimodal perturbation method, i.e., perturbing the learning parameter by weighting the heterogeneous distance metric, the attribute space through the random subspace method and attribute reduction, the training samples by bootstrap sampling and the output target of *k* different neighbors through evidence theory.

We performed experiments on multiple UCI data sets (shown in Table 3) to compare the classification performance of RRSB with other multimodal perturbation-based ensemble algorithms.

To minimize potential inaccuracies caused by the partition of the training set, the results from each data set have been averaged over 50 times. For each experiment we randomly split the data set into two equal-sized sets — the training set (50% of the data) and the test set (the remaining 50%). For all algorithms, the ensemble size was set to 25. For RRSB, the parameters were set as in Section 5.2, that is, $\alpha_0 = 0.95$ and γ's are equal to the inverse of mean distance among the training instances of the corresponding class. $\delta = 0.2$, the random subspace ratio r was randomly set in the range [M/3, 2M/3] for each random subspace, where M denotes the number of attributes in the data set, and k for the number of neighbors was set as 5. Table 7 shows the performances obtained by FASBIR, GAv1, GAv2, EPSO and RRSB, where GAv1 and GAv2 are two different versions of GA-based ensemble algorithms in [17], and EPSO is the PSO-based ensemble algorithm in [18]. To make the experimental results comparable, we did the experiments using the same conditions for all algorithms.

To compare the difference between RRSB and other ensemble algorithms, we selected the paired *t*-test [37] for statistical analysis and set the significance level to 0.05. In the experiments, each of the other ensemble algorithms was paired with RRSB, and a paired *t*-test was employed by using the classification results on different data sets. Table 8 shows the different paired *t*-test results over multiple data sets.

Table 7

Comparison of the proposed algorithm with other algorithms.

Data set	FASBIR	GAv1	GAv2	EPSO	RRSB
sonar	81.76	77.57	76.89	83.65	83.98
ionosphere	84.05	89.43	89.43	93.00	91.54
liver	59.42	65.35	65.81	65.12	65.22
vehicle	69.03	69.91	69.19	70.52	73.51
vowel	87.57	92.24	91.64	85.00	91.29
heart	82.15	77.75	78.61	84.00	81.71
wdbc	95.96	93.17	93.56	95.80	96.73
pima	74.09	74.01	74.24	74.00	77.66
credit-g	72.33	72.36	71.84	73.70	75.81
cardiotocography	100.00	100.00	100.00	100.00	100.00
thoracic	86.11	85.68	86.25	88.31	91.37
diabetic	71.29	66.24	67.31	70.55	76.58
epileptic	55.58	53.38	54.26	59.74	58.75
firm teacher	78.53	77.35	79.54	78.95	83.36
pubchem	98.55	98.59	98.68	98.72	98.76
biodegradation	89.02	87.81	88.24	89.84	92.17
seismic-bumps	95.38	93.15	94.25	93.78	93.82
turkiye student	85.31	85.34	84.89	86.58	88.56
z-alizadeh	88.83	87.25	86.14	86.84	87.51
movement-libras	72.83	77.26	78.59	75.54	83.19
Average	81.39	81.19	81.47	82.68	84.57

It can be seen from Table 7 that FASBIR, GAv1, GAv2, EPSO and RRSB obtained the best accuracies (2, 1, 1, 3 and 12 times, respectively), and the best mean average accuracy was obtained by RRSB. Although RRSB had the lower accuracy on other 7 data sets compared with the other methods, the accuracies of RRSB had small differences compared with the best accuracies, and the biggest difference (2.29%) appeared at the “heart” of the data set.

From Table 8, it can be seen that for each pair of ensemble methods, the *P*-values of different pairs of ensemble algorithms are less than 0.05, i.e., the null hypothesis that the average difference is zero can be rejected. In other words, the difference between RRSB and each of other ensemble methods is statistically significant.

Table 8

Paired *t*-test results

The pair of ensemble methods	*P*-values
RRSB *vs.* FASBIR	0.017
RRSB *vs.* GAv1	0.019
RRSB *vs.* GAv2	0.023
RRSB *vs.* EPSO	0.021

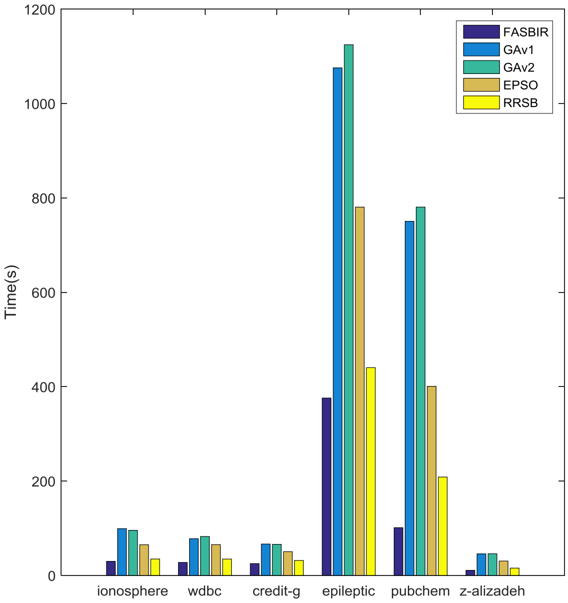


Fig. 2 Testing time on six data sets

Fig. 2 shows the testing time of different methods on 6 data sets, where two of them have hundreds of attributes. The results show that the GA-based methods are the most time consuming, the PSO-based method cost less time than GA, FASBIR requires the shortest time, and the time cost of RRSB is only slightly longer than FASBIR. The reason behind the results is that GA needs much more time for selecting a best attribute subset, especially in high dimensional data sets. Although PSO adopts the random subspace method for selecting an attribute subset, it needs much time to optimize the other parameters. FASBIR and RRSB adopt random subspace and attribute filtering techniques to select attribute subsets, which take less time compared with GA- and PSO-based algorithms.

5.5 Application on network intrusion detection

To test the effectiveness of RRSB on big data set, we used the benchmark of intrusion detection KDD Cup 99 [38] for experiments. The KDD Cup 99 data set is an intrusion detection data set, where each sample in the data set describes a network connection record. Each sample of the data set contains 41 conditional attributes describing the connection records and a class label assigning either normal or attack type to the connection records, where the 41 attributes are divided into 34 numeric attributes and 7 nominal attributes, and all attack types belong to four categories, i.e., PROBE, DOS, U2R and R2L. Since the original data set is too large and contains too many duplicate records, here we use the well-known 10%-KDD Cup 99 data set [38], it contains

494,021 records. The detail of the 10%-KDD Cup 99 data set is described in Table 9.

Table 9
Number of samples for various attack types and normal connections

	Attack categories and normal connections	Original number of records in 10%-KDD Cup 99
PROBE	ipsweep	1,247
	nmap	231
	portsweep	1,040
	satan	1,589
DOS	back	2,203
	land	21
	neptune	107,201
	pod	264
	smurf	280,790
	teardrop	979
U2R	buffer_overflow	30
	loadmodule	9
	perl	3
	rootkit	10
R2L	ftp_write	8
	guess_passwd	53
	imap	12
	multihop	7
	phf	4
	Spy	2
	warezclient	1,020
	warezmaster	20
Normal		97,278
Total		494,021

We computed the detection rate for all ensemble algorithms, where the detection rate is defined as the ratio of the number of detected attack samples to the total number of attack samples. The KDD Cup 99 data set is a representative imbalanced data set [39], and it can be seen from Table 9 that the R2L and U2R types are very rare. The number of attack samples is much less than that of normal samples, but classifying attack samples correctly often has a greater significance than classifying normal samples in intrusion detection. So it is more significant to compute the detection rate for each attack type than to compute the overall detection rate with respect to all connection samples. In the experiments, we computed the detection rate and F-Measure [40] value for each attack type.

In the experiments, we randomly split the data set into two parts (10% of the data for training, and the remaining for testing). The parameters of all algorithms are set the same as those in Section 5.4. The average of 50 times' results are computed, Table 10 and 11 showed the detection rates and F-Measure values of different ensemble methods on 10%-KDD Cup 99 data set, respectively.

From Table 10, it can be seen that for each attack type, RRSB obtained higher detection rate than other ensemble algorithms. In particular, the detection rates on U2R and R2L are significantly higher than those of other ensemble methods, which indicates that RRSB is more effective in detecting rare but important attack categories (U2R and R2L). Moreover, the overall detection rate of RRSB is also higher than those of the other ensemble methods. Table 11 shows that RRSB achieved the highest F-Measure values on "PROBE", "U2R" and "R2L", and the F-Measure value of "DOS" from RRSB is only less than that of FASBIR with 0.04%. This experiment demonstrates that RRSB can be effectively used in big data set. Furthermore, RRSB can obtain good performance on big data set.

Table 10
Detection rate of various attack types

Ensemble	Detection rate for each attack type (%)				Overall detection
method	PROBE	DOS	U2R	R2L	rate (%)
FASBIR	97.79	99.95	59.33	94.82	99.81
GAv1	97.71	99.87	57.14	95.51	99.46
GAv2	96.92	99.89	59.81	95.33	99.36
EPSO	97.98	99.91	61.52	95.17	99.75
RRSB	97.87	99.96	62.41	96.52	99.89

Table 11
F-Measure of various attack types

Ensemble	F-Measure for each attack category (%)			
method	PROBE	DOS	U2R	R2L
FASBIR	90.51	99.56	59.61	81.77
GAv1	91.26	98.97	57.18	83.54
GAv2	90.74	98.86	58.28	82.71
EPSO	92.28	99.04	60.19	82.23
RRSB	93.05	99.51	64.02	85.29

6. Conclusions

In this paper, a novel multimodal perturbation-based ensemble algorithm, RRSB, is proposed, which generates accurate but diverse component classifiers to improve the performance of ensemble classification. The experimental results from multiple UCI data sets show that our proposed method can improve the classification performance in most cases. Compared with other methods, the RRSB is robust, with different *k* values. In addition, the testing time of RRSB is less than GA- and PSO-based methods and is comparable to FASBIR. Finally, the experimental results from the KDD Cup 99 data set show that RRSB is effective for a big and imbalanced data set.

Since the output target of member classifiers only uses majority voting in this paper, in future work, we will study other combination rules on the ensemble. The study of diversity measures [41] among multimodal perturbations is clearly worthy of attention.

Conflict of interest

None.

Acknowledgments

This work was partially supported by National Natural Science Foundation of China (grant number. 61371168).

References

[1] T.G. Dietterich, Machine learning research: four current directions, AI Mag. 18 (4) (1997) 97–136.

[2] Z. Zhou, Y. Yu, Ensembling local learners through multimodal perturbation, IEEE Trans. Syst. Man, Cybern.

Part B Cybern. 35 (4) (2005) 725–735.

[3] T.K. Ho, The random subspace method for constructing decision forests, IEEE Trans. Pattern Anal. Mach. Intell. 20 (8) (1998) 832–844.

[4] T.K. Ho, Nearest neighbors in random subspaces, Joint IAPR Int. Workshops on Adv. in Pattern Recognit. 1451 (1998) 640–648.

[5] J. Gu, L. Jiao, F. Liu, S. Yang, R. Wang, P. Chen, Y. Cui, J. Xie, Y. Zhang, Random subspace based ensemble sparse representation, Pattern Recognit. 74 (2018) 544–555.

[6] J.J. Rodríguez, L.I. Kuncheva, C.J. Alonso, Rotation forest: a new classifier ensemble method, IEEE Trans. Pattern Anal. Mach. Intell. 28 (10) (2006) 1619–1630.

[7] L.I. Kuncheva, J.J. Rodriguez, An experimental study on rotation forest ensembles, in: 7th Int. Work. Mult. Classif. Syst., 2007: pp. 459–468.

[8] W. Siedlecki, J. Sklansky, A note on genetic algorithms for large-scale feature selection, Pattern Recognit. Lett. 10 (1989) 335–347.

[9] L. Breiman, Bagging Predictors, Mach. Learn. 24 (2) (1996) 123–140.

[10] Y. Freund, R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, J. Comput. Syst. Sci. 55 (7) (1997) 119–139.

[11] G.P. Zhang, Neural networks for classification: a survey, IEEE Trans. Syst. Man Cybern. Part C (Applications Rev). 30 (4) (2000) 451–462.

[12] B. Gabrys, D. Ruta, Genetic algorithms in classifier fusion, Appl. Soft Comput. 6 (4) (2006) 337–347.

[13] D. Kocev, C. Vens, J. Struyf, S. Džeroski, Tree ensembles for predicting structured outputs, Pattern Recognit. 46 (3) (2013) 817–833.

[14] J. Tian, M. Li, F. Chen, J. Kou, Coevolutionary learning of neural network ensemble for complex classification tasks, Pattern Recognit. 45 (4) (2012) 1373–1385.

[15] Y. Bao, N. Ishii, Combining multiple k-nearest neighbor classifiers using different distance functions, Int. Conf. on Data Eng. Autom. Learn. 3177 (2004) 634–641.

[16] N. Ishii, E. Tsuchiya, Y. Bao, N. Yamaguchi, Combining classification improvements by ensemble processing, in: Proc. Third ACIS Int. Conf. Softw. Eng. Res. Manag. Appl. SERA 2 2005: pp. 240–246.

[17] H. Altinçay, Ensembling evidential k-nearest neighbor classifiers through multi-modal perturbation, Appl. Soft Comput. 7 (3) (2007) 1072–1083.

[18] L. Nanni, A. Lumini, Particle swarm optimization for ensembling generation for evidential k-nearest-neighbour classifier, Neural Comput. Appl. 18 (2) (2009) 105–108.

[19] Z. Pawlak, Rough sets, Int. J. Comput. Inf. Sci. 11 (5) (1982) 341–356.

[20] Y.Y. Yao, Relational interpretations of neighborhood operators and rough set approximation operators, Inf. Sci. 111 (1-4) (1998) 239–259.

[21] Q. Hu, D. Yu, Z. Xie, Neighborhood classifiers, Expert Syst. Appl. 34 (2) (2008) 866–876.

[22] Q. Hu, D. Yu, J. Liu, C. Wu, Neighborhood rough set based heterogeneous feature subset selection, Inf. Sci. 178 (18) (2008) 3577–3594.

[23] T. Denoeux, A k-nearest neighbor classification rule based on Dempster-Shafer theory, IEEE Trans. Syst. Man. Cybern. 25 (5) (1995) 804–813.

[24] P.J.F. Groenen, K. Jajuga, Fuzzy clustering with squared Minkowski distances, Fuzzy Sets Syst. 120 (2) (2001) 227–237.

[25] C. Stanfill, D. Waltz, Toward memory-based reasoning, ACM Commun. 29 (12) (1986) 1213–1228.

[26] D.R. Wilson, T.R. Martinez, Improved heterogeneous distance functions, J. Artif. Intell. Res. 6 (1) (1997) 1–34.

[27] J.Z. Huang, M.K. Ng, H. Rong, Z. Li, Automated variable weighting in k-means type clustering, IEEE Trans. Pattern Anal. Mach. Intell. 27 (5) (2005) 657–668.

[28] C. Lian, S. Ruan, T. Denœux, An evidential classifier based on feature selection and two-step classification strategy, Pattern Recognit. 48 (7) (2015) 2318–2327.

[29] E.M. Kleinberg, Stochastic discrimination, Ann. Math. Artif. Intell. 1 (1-4) (1990) 207–239.

[30] X. Luo, Y. Xia, Q. Zhu, Y. Li, Boosting the k-nearest-neighborhood based incremental collaborative filtering, Knowledge-Based Syst. 53 (2013) 90–99.

[31] J. Calvo-Zaragoza, J.J. Valero-Mas, J.R. Rico-Juan, Improving kNN multi-label classification in Prototype Selection scenarios using class proposals, Pattern Recognit. 48 (2015) 1608–1622.

[32] T. Prasartvit, A. Banharnsakun, B. Kaewkamnerdpong, T. Achalakul, Reducing bioinformatics data dimension with ABC-kNN, Neurocomputing. 116 (2013) 367–381.

[33] N. García-Pedrajas, D. Ortiz-Boyer, Boosting k-nearest neighbor classifier by means of input space projection, Expert Syst. Appl. 36 (2009) 10570–10582.

[34] M. Lichman, UCI Machine Learning Repository, 2018. <http://archive.ics.uci.edu/ml>.

[35] L.I. Kuncheva, Combining pattern classifiers: methods and algorithms, John Wiley and Sons, 2014.

[36] R. Maclin, D. Opitz, Popular ensemble methods: an empirical study, J. Artif. Intell. Res. 11 (1999) 169-198.

[37] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7(1) (2006) 1-30.

[38] KDD Cup 99 Dataset, 1999. <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>.

[39] J. Liu, Q. Hu, D. Yu, A weighted rough set based method developed for class imbalance learning, Inf. Sci. 178 (4) (2008) 1235–1256.

[40] I. Pillai, G. Fumera, F. Roli, Designing multi-label classifiers that maximize F-measures: state of the art, Pattern Recognit. 61 (2017) 394–404.

[41] G.D.C. Cavalcanti, L.S. Oliveira, T.J.M. Moura, G. V. Carvalho, Combining diversity measures for ensemble pruning, Pattern Recognit. Lett. 74 (2016) 38–45.

[42] J. Liang, Q. Hu, P. Zhu, W. Wang, Efficient multi-modal geometric mean metric learning, Pattern Recognit. 75 (2018) 1339–1351.

[43] D. Zhai, X. Liu, H. Chang, Y. Zhen, X. Chen, M. Guo, W. Gao, Parametric local multiview hamming distance metric learning, Pattern Recognit. 75 (2018) 250–262.

[44] X. Lv, F. Duan, Metric learning via feature weighting for scalable image retrieval, Pattern Recognit. Lett. 109 (2017) 97-102.

[45] B. Nguyen, C. Morell, B. De Baets, Supervised distance metric learning through maximization of the Jeffrey divergence, Pattern Recognit. 64 (2017) 215–225.

[46] H. Wang, L. Feng, X. Meng, Z. Chen, L. Yu, H. Zhang, Multi-view metric learning based on KL-divergence for similarity measurement, Neurocomputing. 238 (2017) 269–276.

Youqiang Zhang, born in 1990, received the M.S. degree in software engineering from Qingdao University of Science and Technology, Qingdao, China. He is currently pursuing the Ph.D. degree in computer science and technology at Nanjing University of Science and Technology, Nanjing, China. His research interests include machine learning, rough sets, and remote sensing image processing.

Guo Cao, born in 1977, received the Ph.D. degree in pattern recognition and intelligence system from Shanghai Jiao Tong University (SJTU), Shanghai, China, in 2006. He has been an Associate Professor with the Department of Computer Science, Nanjing University of Science and Technology, Nanjing, China, since 2010. His research interests include machine learning, remote sensing image processing and biometrics.

Bisheng Wang, received the B.S. degree in computer science and technology from Nanjing University of Science and Technology (NUST), Nanjing, China, in 2016. He is currently pursuing the Ph.D. degree in computer science and technology at NUST. His research interests include deep learning, image processing and object detection.

Xuesong Li, received the B.S. degree in computer science and technology from Nanjing University of Science and Technology (NUST), Nanjing, China, in 2014. He is currently pursuing the Ph.D. degree in computer science and technology at NUST. His research interests include image processing and sparse representation.