International Journal of Approximate Reasoning 187 (2025) 109561

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

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

Explainable granular fusion: Graph-embedded rectangular neighborhood rough sets for knowledge system convergence

Yigao Li, Weihua Xu *

College of Artificial Intelligence, Southwest University, Chongqing 400715, PR China

ARTICLE INFO

Keywords:
Information fusion
Granular-rectangular neighborhood rough sets
Graph theory
Binary tree

ABSTRACT

With the development of Rough Set Theory (RST), many improved theories based on RST have emerged. Some of these theories have been applied in the field of feature selection, significantly improving its efficiency. However, they have not yet been widely used in multi-source information domains. This paper proposes a multi-source information fusion method based on Granular-Rectangular Neighborhood Rough Set (GRNRS) and graph theory. First, an improved algorithm based on GRNRS is proposed to evaluate the contribution of each information source to a classification task under a specific attribute. In this process, we provided rigorous theoretical proofs for the concepts and mechanisms used in the improved GRNRS. Meanwhile, the Pearson Correlation Coefficient (PCC) is used to assess the linear relationship between information sources. Then, by integrating the results of the improved GRNRS algorithm and PCC, the adjacency matrix of a graph is constructed. Finally, the preference value of each information source under a specific attribute is calculated based on the adjacency matrix. Information fusion under a specific attribute is achieved by selecting the information source with the highest preference value. Extensive experiments are conducted to analyze the impact of the algorithm's parameters on its final performance. Meanwhile, our method is compared with seven other information fusion algorithms using three metrics: classification accuracy, Average Quality (AQ), and runtime. Friedman and Nemenyi tests are conducted on the comparison results under the classification accuracy and AQ metrics, demonstrating that there are significant differences among the algorithms. The results demonstrate that the proposed algorithm is both time-efficient and effective.

1. Introduction

The advent of the information age has highlighted the increasing significance of data in daily life. With the development of the internet, information transmission has become faster, and the channels for obtaining information have become more diverse. Consequently, extracting useful information from vast sources has become a critical challenge. In the late 1970s, the concept of multi-source information fusion emerged. This concept refers to the process of integrating information from multiple sources into a single, unified dataset. Following its introduction, numerous theories and algorithms have been developed. After 2010, research in this field began to grow exponentially, and today, multi-source information fusion has become a highly popular research area. Multi-source information fusion has applications in various fields, such as health [21][20], economy [11][35], social networks [5][7], pattern recognition [43][17][29] and so on. Next, we will review the relevant theories in this field in detail.

* Corresponding author.
E-mail addresses: lyg040826@email.swu.edu.cn (Y. Li), chxuwh@gmail.com (W. Xu).

https://doi.org/10.1016/j.ijar.2025.109561
Received 11 May 2025; Received in revised form 14 August 2025; Accepted 20 August 2025

Available online 28 August 2025

Some classical fusion strategies used by multi-source information fusion are listed below. The first category includes probability model-based fusion methods. This type of method primarily evaluates the quality of each information source through probabilistic models. Most approaches use Bayesian probabilistic models for the evaluation. For example, Cheng [4] proposed a multi-source data fusion method based on Bayesian inference (BIF), which constructs a likelihood function using two error models. The second category consists of methods based on Belief Function Theory (BFT). Two classic examples of this theory are Dempster-Shafer Theory (DST) and Dezert-Smarandache Theory (DSmT). For example, Fei [9] has established a thorough assessment indicator system and proposed an innovative evaluation framework. This framework incorporates a BPA model, a new BPA generation method for observed values, and sophisticated hierarchical weighting and fusion techniques. Yaghoubi [32] introduces a novel metric as the basis for a preprocessing technique designed to assess and reduce the conflict among the evidences. Dong [6] develops a new sensor fusion strategy for HAR in BSNs using DSmT, which significantly enhances recognition accuracy. Boumediene [2] presents a DSmT-based evidential data association method that highlights crucial information and filters out unreliable data. This kind of method also includes [8][13]. Some methods are based on the Fuzzy Set Theory (FST). FST breaks through the binary limitation of belonging or not belonging in classical set theory. In this framework, an element can belong to a set with a certain probability, which greatly enhances the flexibility of the information source evaluation system. For example, Cai [3] constructs fuzzy similarity relations using Kullback–Leibler divergence and develops a fuzzy decision-making model, introducing fuzzy approximate conditional entropy and an entropy fusion model. Mao [15] represents the Pythagorean fuzzy set (PFS) within a belief structure framework and quantifies its uncertainty using fractal-based belief (FB) entropy. Zhou [42] proposes a novel approach using the Intuitive Fuzzy Cloud Model (IFCM) to generate more precise BPAs by comprehensively quantifying information uncertainty. There are also many information fusion algorithms based on fuzzy set theory (FST), such as [23][22]. The fourth kind of strategy is Rough Set Theory (RST). RST was proposed by Pawlak [16] in 1998 and has been widely applied in the field of multi-source information fusion. Zhang [38] and Wei [26] summarize methods based on RST for multi-source data of different types. Zhang [37] overviews the neutrosophic fusion method based on RST.

Next, we will elaborate on the information fusion methods based on RST. In RST, various methods for generating rough sets and evaluating information sources using these rough sets lead to different information fusion approaches. For example, Li [12] calculates a threshold using the variance and mean of all sample values in the attributes, then generates a rough set based on this threshold and derives information entropy to achieve multi-source information fusion. With the development of RST-based information fusion methods, some multi-source information fusion approaches have integrated RST with other methods, significantly improving the efficiency of information fusion. For instance, Sang [18] applies the Fuzzy Rough Set theory for information fusion, which combines FST and RST. This approach improves RST in aspects such as rough set construction, equivalence class computation, and the calculation of upper and lower approximation sets, making it applicable to fuzzy sets and enhancing the efficiency of information fusion. There are more methods apply RST to information fusion task, such as [14][36][33].

With the development of RST, various improvements to the theory itself have emerged to expand its applicability and enhance its efficiency. Wang [25] proposes the local neighborhood rough set, which integrates neighborhood and local rough sets to effectively handle large-scale data with limited labels. Al-shami [1] introduce the subset neighborhood under an arbitrary binary relation and define corresponding approximations, accuracy, and roughness measures. Some theories have applied the improved RST to feature selection tasks. A Weighted Neighborhood Rough Set model is proposed by Hu [10] using weighted neighborhood relations, with a dependency measure to assess attribute significance and a greedy algorithm for attribute selection. To address noise sensitivity in neighborhood rough sets, the Weighted k-Nearest Neighborhood Rough Set model is proposed by Wang [24], incorporating class label standard deviation to weight neighbor samples and evaluate sample quality. There are also many papers on using rough set theory for feature selection tasks, such as [34][19][41]. These theories have significantly improved the efficiency of feature selection. However, most of the improved RST theories have not been applied to the field of information fusion. So we apply the Granular-Rectangular Neighborhood Rough Set (GRNRS) [28], which is one of the improvements of RST, for the field of information fusion and propose an optimization method based on GRNRS. Considering that most information fusion algorithms do not account for the relationships between different information sources, we introduce correlation coefficients to evaluate the relationships among them. To obtain a more accurate assessment of the relationships between the quality of different information sources, we incorporate graph theory. By constructing a graph, we can comprehensively consider the relationships between each pair of information sources and derive the overall quality relationship. The contributions of this paper are listed below:

- We integrate RST and GRNRS theories to propose a model for evaluating the role of each information source in a classification task under a single attribute. First, we use a partition binary tree to generate a neighborhood set. Then, we evaluate the contribution of each information source to the classification task using neighborhood approximate accuracy, which can be calculated by the lower approximation set. In this process, we propose an improved theory based on GRNRS to enhance computational efficiency. This improved theory has less runtime and provides a partial reference for the information fusion in subsequent processes.
- We evaluate the linear correlation between two information sources using the Pearson Correlation Coefficient (PCC). Then, based on the results of PCC and the improved GRNRS, we construct the adjacency matrix of the graph. This approach considers both the contribution of each information source to the classification task and the correlation between information sources. This graph adjacency matrix will serve as the basis for evaluating the quality of each information source under specific attributes.
- To more effectively extract information from the graph adjacency matrix, we proposed a method for calculating the preference value vector based on graph theory. It allows us to select the information source with the highest preference value as the output under a specific attribute. In this process, we propose an estimation algorithm for computing the preference value, which improves computational efficiency while ensuring result accuracy.
- We conducted experiments to analyze the impact of all parameters in our proposed method on the final algorithm performance, as well as experiments to explore the effectiveness and time efficiency of our proposed method. Friedman and Nemenyi tests are

Table 1
An example of MSDIS.

	$f^1_{a_1}$	$f^1_{a_2}$	$f^1_{a_3}$	$f^1_{a_4}$	$f^2_{a_1}$	$f^2_{a_2}$	$f^2_{a_3}$	$f^2_{a_4}$	$f^3_{a_1}$	$f^3_{a_2}$	$f^3_{a_3}$	$f^3_{a_4}$	f_d
x_1	5.1	3.5	1.4	0.2	5.2	3.3	1.6	0.1	5.4	3.6	1	0	1
x_2	4.9	3	1.4	0.2	5	3.4	1.1	0.3	5.2	2.6	1.8	0.5	1
x_3	4.7	3.2	1.3	0.2	4.9	3.1	1.1	0.4	5.1	3.1	0.9	0	1
x_4	7	3.2	4.7	1.4	7.2	2.9	4.7	1.3	7.1	3.4	4.4	0.9	2
x_5	6.4	3.2	4.5	1.5	6	3.4	4.5	1.1	6.2	2.8	4.7	1.3	2
x_6	6.9	3.1	4.9	1.5	6.8	2.6	5.3	1.5	6.5	3.1	4.9	1.9	2
x_7	6.3	3.3	6	2.5	6.6	2.8	6.4	2.2	6.1	3.3	5.6	2.9	3
x_8	5.8	2.7	5.1	1.9	5.9	2.5	5.4	1.6	5.4	3.1	4.9	2.1	3
x_9	7.1	3	5.9	2.1	7	2.7	6	2	7.2	3.3	6.1	2	3

conducted on the comparison results under the classification and AQ metrics, demonstrating that there are significant differences among the algorithm. At the same time, we conducted an in-depth analysis of the experimental results to derive conclusions regarding parameter selection and the advantages and disadvantages of our proposed method.

Our proposed method also has certain limitations that require further research and improvement.

• Our algorithm performs poorly on datasets with many attributes but few samples. To address this issue, it is necessary to improve the evaluation metrics for information sources within the algorithm, thereby enhancing its adaptability to such data scenarios.

• Our algorithm is currently limited to handling single-valued multi-source information systems. However, there are other types of multi-source information systems, such as interval-valued and multi-scale systems. To extend the applicability of our algorithm to these different types, it is necessary to modify the evaluation method for information sources, enabling the algorithm to adapt to various forms of multi-source data.

Next, we provide a brief introduction to the overall structure of the paper. In section 2, we review relevant knowledge on multi-source decision information systems, RST and GRNRS, PCC, and graph theory. In section 3, we propose the information fusion method based on GRNRS and graph theory. Additionally, we present a schematic diagram and a pseudo-algorithm of the proposed method in this section. In section 4, we provide details on datasets, computational hardware configurations, and experimental design, followed by an analysis of the experimental results. In section 5, we summarize the proposed method and the conclusions drawn from the experiments. We also discuss the limitations of our method and suggest directions for future research.

2. Preliminaries

In this section, we will briefly review the conception of MSDIS, granular-rectangular neighborhood rough sets and graph theory. At the same time, we also introduce the relationships between the conceptions and their roles in the theory we propose.

2.1. Multi-source decision information system

Before introducing MSDIS, let's first introduce the basic unit that makes up MSDIS, which is DIS. DIS is an information system that records the values of various samples under different attributes. We refer to the collection of these samples as U, and the collection of attributes as A. Each sample corresponds to a value under each attribute, and these values form a set of values V. The mapping relationship between samples and attributes that generates the values is called $f_A : U \times A \to V$. These elements form an IS, but the difference between DIS and IS is that DIS not only has an attribute set A but also has a decision attribute d. Thus, we provide the mathematical definition of DIS and MSDIS.

Definition 1. $DIS = (U, A, d, f_A, V_A, f_d, V_d)$, where U, A, d separately represent sample set, attribute set, decision attribute. V_A represents the value set of attribute set A. V_d represents the value set of decision attribute d. $f_A = U \times A \to V_A$ and $f_d = U \times d \to V_d$. MSDIS is a collection of multiple DIS, representing data collected from multiple information sources for these samples. $MSDIS = \{DIS_k | DIS_k = (U, A, d, f^k_A, V^k_A, f^k_d, V^k_d), k = 1, 2, ..., l\}$. An example of MSDIS is shown in Table 1.

2.2. Rough set theory and granular-rectangular neighborhood rough set

Rough set theory is proposed by Pawlak [16], this theory is used to measure whether an attribute column in a DIS is suitable for classification tasks in this paper. Mathematically, we define any subset R of the product of the domain $U \times U = \{(a, b) | a, b \in U\}$ as a relation. If the relation R satisfies the following three conditions, then R is an equivalence relation.

• For any $a \in U$, $(a, a) \in R$.

• If $(a, b) \in R$, then $(b, a) \in R$.

• If $(a, b) \in R$ and $(b, c) \in R$, then $(a, c) \in R$.

For any $a \in U$, we define the set of all elements that are equivalent to a as an equivalence class $[a]_R = \{b \in U | (a, b) \in R\}$. These equivalence classes form a partition of U, denoted as U/R. An equivalence relation R can form exactly one partition in U. If the set X can be represented as the union of several sets from the partition U/R, then X is called a precise set; otherwise, it is called a rough set. Rough set theory is developed to address a series of issues related to rough sets.

To define the roughness of a set X, rough set theory defines the upper approximation set and the lower approximation set. For a set X, the formulas for the upper and lower approximation sets under the partition U/R are as follows:

$$\underline{R(X)} = \cup\{Y \in U/R | Y \subseteq X\} \tag{1}$$

$$\overline{R(X)} = \cup\{Y \in U/R | Y \cap X \neq \emptyset\}. \tag{2}$$

Subsequently, we can calculate the roughness of a set using the approximation accuracy, which is defined as follows:

$$AP_{U/R}(X) = \frac{|\underline{R(X)}|}{|\overline{R(X)}|}, \tag{3}$$

where $|*|$ represent the amount of unit in the set. This accuracy lies between 0 and 1. The greater the accuracy of set X, the more precise the set is; Conversely, the lower the accuracy, the rougher the set is. If the accuracy of set X is 1, then X is called a precise set; Otherwise, it is called a rough set.

Next, we introduce the neighborhood rough set. This theory is an extension of rough set theory, continuing the idea of calculating the roughness of set X through the upper and lower approximation sets. However, it abandons the original method of constructing a partition of the entire universe U using equivalence relations, and instead introduces the concept of neighborhood. Next, we provide the definition of a neighborhood S_t in the context of a decision information system.

Definition 2. For a $DIS = (U, A, d, f_A, V_A, f_d, V_d)$, neighborhood set $S = \{S_1, S_2, ..., S_t, ..., S_q\}$, where $S_t \subseteq U, t = 1, 2, ..., q$. A neighborhood S_t contains elements that are similar to each other within U.

The difference between a neighborhood S_t and the sets in the rough set partition U/R is that elements within a neighborhood can be repeated, while in rough set theory, for any $X, Y \in U/R$, there are no repeated elements between X and Y. There are many different methods for generating neighborhoods, which has led to various derivative approaches in different directions.

Granular-Ball Neighborhood Rough Set (GBNRS) [27] method is one of these approaches. GBNRS uses granular balls to represent both upper and lower approximations, achieving a unified approach for handling both discrete and continuous data. A granular ball is composed of a set of sample points, with each ball having a center and radius. This method provides multi-scale learning capabilities and offers stronger robustness and adaptability when processing data.

However, this method can be time-consuming. To reduce the computation time, the Granular-Rectangular Neighborhood Rough Set (GRNRS) [28] is proposed. This method introduces a binary tree to generate neighborhoods, with the binary tree storing neighborhoods of different scales. By calculating the distances between nodes in the binary tree, the final neighborhood set is obtained. This approach not only reduces computation time but also ensures robustness and adaptability, while also providing a unified approach for handling both discrete and continuous data. GRNRS method is used to determine whether each DIS in the input MSDIS is suitable for classification data. It will be part of evaluating the quality of a specific source. This method will be discussed in detail in subsection 3.1 and subsection 3.2.

2.3. Pearson correlation coefficient

To evaluate the correlation between two sources, we introduce the Pearson Correlation Coefficient (PCC). The PCC is used to calculate the linear correlation between two variables. Below is the definition of the PCC.

Given two variables X and Y, the PCC is defined as the ratio of the covariance between the two variables to the product of their respective standard deviations, its formula is shown as follows:

$$r = \frac{cov(X,Y)}{\rho_X \rho_Y} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{(n\sum x_i^2 - (\sum x_i)^2)}\sqrt{(n\sum y_i^2 - (\sum y_i)^2)}}, \tag{4}$$

where $cov(X,Y)$ represents the covariance between X and Y, ρ_X and ρ_Y separately denote respective standard deviations of X and Y, n is the number of samples in the variable, x_i and y_i separately mean the value of i-th sample in X and Y.

The value of the PCC ranges from -1 to 1. The closer the absolute value is to 1, the stronger the linear correlation between the two variables. If the absolute value is 1, it indicates that the values of the two variables' corresponding samples lie on a straight line. Compared with Spearman Correlation Coefficient (SCC), it can not only spend less time for discarding the procedure of calculating rank of every sample in the variable, but also achieve the same effect in the information fusion task. If a information source has a higher linear correlation with other sources in terms of a specific attribute, it means that the source can better represent the other sources in that attribute. Therefore, we choose the PCC as part of the evaluation of the quality of a source. The specific application method will be explained in detail in subsection 3.3.

2.4. Graph theory

To select the best information source based on the preference values between any two information sources, we use graph theory methods. In this section, we will review the graph theory.

Definition 3. An undirected graph G can be mathematically represented as an ordered pair $< V, E >$, where V is the set of vertices, and the elements in this set are the vertices of the graph. E is the set of edges, and the elements in this set are pairs from $V \times V$, representing the binary relationships between vertices. If every vertex in an undirected graph G is adjacent to all other vertices, then G is called a complete undirected graph. If each edge e in the edge set E of the graph G has an associated weight $w(e)$, then the graph G is called a weighted graph.

For a complete undirected weighted graph, it can be represented by a matrix A, where each element $A[i][j]$ represents the weight of the edge between vertex i and vertex j. This matrix is typically symmetric because the graph is undirected, meaning $A[i][j] = A[j][i]$. In graph theory, a path is a sequence of vertices such that each adjacent pair of vertices in the sequence is connected by an edge in the graph. Formally, a path can be represented as: $\Gamma = v_0 v_1 v_2 \dots v_k$, where for every $i = 0, 1, \dots, k-1$, there exists an edge (v_i, v_{i+1}) in the graph. The method for selecting the best source using the adjacency matrix will be explained in detail in subsection 3.4.

3. Information fusion method based on GRNRS and graph theory

In this section, we will provide a detailed introduction to the granular-rectangle neighborhood rough sets theory and its application in information fusion. Additionally, we will introduce the application of graph theory in information fusion.

3.1. Neighborhood set

In this subsection, we will introduce the method of generating neighborhood set using GRNRS. In MSDIS, for each attribute a in every DIS_k, a neighborhood set S_k^a can be generated. This will be used later to evaluate the effect of each DIS under specific attribute a to classification task. Now given a $DIS_k \in MSDIS$ and an attribute $a \in A$, we need to firstly sort each sample u in ascending order based on the value of its attribute a under the DIS_k. After that, a partition binary tree will be generated based on the sorted sample set U_{sorted}.

Definition 4. A partition binary tree T_k^a is a special type of binary tree in which each node stores a set of elements. The root node stores the entire initial set $U = \{u_1, u_2, ..., u_n\}$, and each internal node divides its stored set into two parts. Defining the first half U_l and the second half U_r of the set U as follows:

If n is even, let $n = 2m$, then:

$$U_l = \{u_i \mid 1 \le i \le m\}, \quad U_r = \{u_i \mid m+1 \le i \le n\}. \tag{5}$$

If n is odd, let $n = 2m+1$, then:

$$U_l = \{u_i \mid 1 \le i \le m\}, \quad U_r = \{u_i \mid m+1 \le i \le n\}. \tag{6}$$

Next, storing U_l and U_r in its left and right child nodes. This partitioning process is recursively applied to the child nodes until all leaf nodes contain only a single element. Leaf nodes store individual elements from the set and are not further subdivided.

Through Definition 4, we can generate a partition binary tree T_k^a with a sorted sample set U_{sorted}. The partition binary tree defined in Definition 4 has some properties which will be stated in Proposition 2.

Proposition 1. *Any partition binary tree T_k^a is a balanced binary tree, meaning that the height difference between the left and right subtrees of any node is no greater than 1.*

Proof 1. For any node storing a set $U = \{u_1, u_2, \dots, u_n\}$, the set is divided into two subsets U_l and U_r. If n is even, $n = 2m$, then the subsets are defined as $U_l = \{u_i \mid 1 \le i \le m\}$ and $U_r = \{u_i \mid m+1 \le i \le n\}$. If n is odd, $n = 2m+1$, then $U_l = \{u_i \mid 1 \le i \le m\}$ and $U_r = \{u_i \mid m+1 \le i \le n\}$. In both cases, the sizes of U_l and U_r differ by at most one. This partitioning process is applied recursively to each subset, creating left and right child nodes, until each leaf node contains a single element. As each division maintains nearly equal subset sizes, the resulting binary tree ensures that the height difference between the left and right subtrees of any node is at most 1. This meets the requirement for a balanced binary tree, such as an AVL tree, and guarantees that the tree maintains a logarithmic height structure.

Proposition 2. *For any node N in the partition binary tree T_k^a, the height of its subtree T_{sub} satisfies $height(T_{sub}) = \lceil \log_2(n) \rceil + 1$, where $\lceil x \rceil$ represents the ceiling function, returning the smallest integer greater than or equal to x. n means the number of elements in the set contained in the node N.*

Proof 2. From the properties of a binary tree, we can deduce that for a full binary tree of height h, the number of leaf nodes is $2^{(h-1)}$. According to Definition 4, the number of elements n in any node N corresponds to the number of leaf nodes in its subtree T_{sub}. Thus, for a full partition binary tree, we have $n = 2^{(h-1)}$, where n is the number of elements in the root node. Now, consider a partition binary tree of height h, whose number of nodes lies between a full partition binary tree of height $h-1$ and height h, we have $2^{(h-2)} < n \le 2^{(h-1)}$.

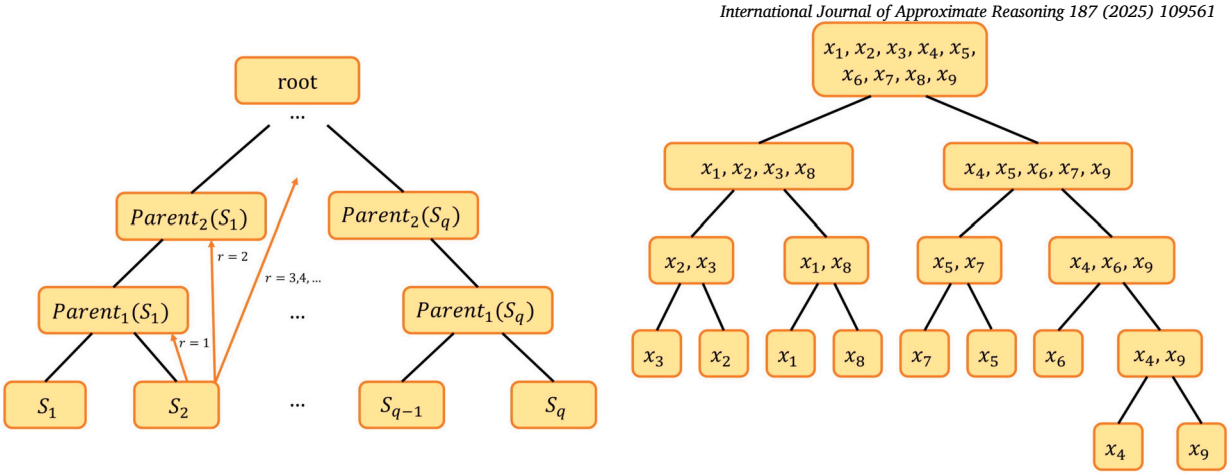

Fig. 1. (a) The illustration of the binary partition tree and neighborhood radius. (b) The example of binary partition tree.

Taking the natural logarithm on both sides, we get $h-2 < \log_2(n) \leq h-1$, which leads to $\log_2(n) \leq h-1 < \log_2(n)+1$. Since $h-1$ is an integer, we obtain $h-1 = \lceil \log_2(n) \rceil$. Thus, solving for $h = \lceil \log_2(n) \rceil + 1$.

After that, the concept of neighborhood radius r is introduced to generate the neighborhood set. The neighborhood radius describes the size of a neighborhood. The larger the value of r, the more elements the neighborhood contains.

Definition 5. For a leaf node l in a partition binary tree, when the neighborhood radius $r = p$, its neighborhood is defined as the value of the ancestor node at the p-th level upward along the tree. It can be represented as the neighborhood $S_t = value(Parent_r(l))$, where $value(*)$ means the value of node $*$.

Using Definition 5, we can calculate a neighborhood S_t with a leaf node l. The neighborhood set S_k^a is the collection of neighborhoods generated by all leaf nodes of the partition binary tree T_k^a based on the neighborhood radius. An illustration of the binary partition tree and the neighborhood radius is shown in Fig. 1(a). All nodes with a neighborhood radius of r share common properties, which we will describe in Proposition 3.

Proposition 3. *For any node N in a partition binary tree T_k^a that contains a neighborhood $S_t = \{x_0, x_1, \ldots, x_n\}$ with a neighborhood radius of r, it must satisfy one of the following two conditions:*

1. $\lceil \log_2(n) \rceil = r$
2. $\lceil \log_2(n) \rceil = r+1$ *and* $\log_2(n)$ *is not an integer.*

Proof 3. According to Definition 5, we can conclude that a node with a neighborhood radius of r is the rth parent of a leaf node. Consequently, the height of its subtree T_{sub} is at least $r+1$. For a full partition binary tree's subtree T_{sub} with height $r+1$, all its leaf nodes are distributed at the bottom level. Thus, for all leaf nodes, the root node of this tree stores a neighborhood that satisfies the neighborhood radius condition of r. Otherwise, subtree T_{sub}'s leaf nodes are distributed across the last two levels. For all the leaf nodes at the bottom level, the root node of this tree stores a neighborhood satisfying the neighborhood radius condition of r. For the leaf nodes in the second-to-last level, the parent node of the root stores a neighborhood satisfying the neighborhood radius condition of r. Therefore, we conclude that a node N storing a neighborhood radius of r must satisfy one of the following two conditions in its subtree T_{sub}: $h = r+1$; $h = r+2$ and the tree is not full, where h means the height of subtree T_{sub}. For the first condition, according to Proposition 2, the height of the subtree rooted at node N is $\lceil \log_2(n) \rceil + 1$. Therefore, we have the equation $\lceil \log_2(n) \rceil + 1 = r+1$. Simplifying this, we obtain $\lceil \log_2(n) \rceil = r$. For the second condition, since a full partition binary tree always satisfies $n = 2^m$ for some integer m, to ensure that T_{sub} is not a full partition binary tree, we only need to guarantee that for any integer m, $n \neq 2^m$. This condition can be simplified as: $\log_2(n)$ is not an integer. Following the same reasoning as in the first condition for subtree height, we get $\lceil \log_2(n) \rceil + 1 = r+2$, which simplifies to $\lceil \log_2(n) \rceil = r+1$.

By applying the above method, we can generate a neighborhood set S_k^a for each DIS_k under a specific attribute a. To make the computation of the neighborhood set easier to understand, we provide a relevant example in Example 3.1.

Example 3.1. In the example, we use a subset of the Iris data from the UCI dataset, as shown in Table 1. Taking a_1 of the DIS_1 as an example, we first need to sort the samples in ascending order based on their attribute values. The resulting sample order is: $x_3 < x_2 < x_1 < x_8 < x_7 < x_5 < x_6 < x_4 < x_9$. Afterward, we generate a binary partition tree based on this sequence. The generated binary partition tree is shown in Fig. 1(b). Suppose we need to determine the neighborhood set composed of all neighborhoods with

a neighborhood radius $r = 1$. We perform a depth-first traversal and select the neighborhoods from the nodes that satisfy the two conditions specified in Proposition 3. These neighborhoods collectively form the neighborhood set S. Let's start from the root of the tree, $\lceil \log_2(n) \rceil = \lceil \log_2(9) \rceil = 4 \neq r$. Next, we examine the left child of the root node, $\lceil \log_2(n) \rceil = \lceil \log_2(4) \rceil = 2 = r + 1$, but $\log_2(4) = 2$ is an integer. We then continue to examine its left child, $\lceil \log_2(n) \rceil = \lceil \log_2(2) \rceil = r$. Thus, we add the neighborhood $\{x_2, x_3\}$ stored in that node to the neighborhood set S. By continuing the depth-first traversal of the binary partition tree, we can obtain the final neighborhood set $S = \{\{x_2, x_3\}, \{x_1, x_8\}, \{x_5, x_7\}, \{x_4, x_6, x_9\}, \{x_4, x_9\}\}$.

The algorithm for generating a neighborhood set S using a constructed partition binary tree T_k^a is presented in Algorithm 1. The time complexity of the algorithm is $O(m)$, where m denotes the number of samples. Next, we will introduce how to evaluate the effect of each DIS_k under specific attribute a to classification task.

Algorithm 1: The algorithm for generating neighborhood set using a constructed partition binary tree.

```
Function FindNeighborhood (N, r, S)
    Input: Node N, Neighborhood radius r, Neighborhood set S
    Output: Neighborhood set S
    if |N.value| = 1 then
        return S
    end
    else if ⌈log2(|N.value|)⌉ = r then
        S.append(value(N))
        return S
    end
    else if ⌈log2(|N.value|)⌉ = r + 1 and log2(|N.value|) is not an integer then
        S.append(value(N))
        FindNeighborhood(N.left, r, S)
        FindNeighborhood(N.right, r, S)
    end
    else
        FindNeighborhood(N.left, r, S)
        FindNeighborhood(N.right, r, S)
    end
End Function

Function Main (T_k^a, r)
    Input: Partition binary tree T_k^a, Neighborhood radius r
    Output: Neighborhood set S
    N ← T_k^a.root
    S ← ∅
    return FindNeighborhood (N, r, S)
End Function
```

3.2. Neighborhood approximate accuracy

In this subsection, we will introduce a method to evaluate each DIS_k under a specific attribute a. For an MSDIS, we can divide the samples into multiple sets based on their categories, forming a partition in the universal set U, denoted as $U/d = \{X_1, X_2, \ldots, X_h, \ldots, X_p\}$, where p represents the number of categories. According to Equation (1), we define the lower approximation of set X_h under neighborhood set S_k^a as follows:

$$\underline{NR_k^a(X_h)} = \cup\{Y \in S_k^a | Y \subseteq X_h\}. \tag{7}$$

Using Equation (7), we can compute the lower approximation set for each category's sample set. To account for all categories, we need to merge the lower approximation sets of all category samples. Then, we have

$$\underline{GNR_k^a(U)} = \cup \underline{NR_k^a(X_h)}, h = 1, \ldots, p. \tag{8}$$

Thus, we present the calculation formula for neighborhood approximate accuracy as follows:

$$NAP^a(DIS_k) = \frac{|\underline{GNR_k^a(U)}|}{|U|}, \tag{9}$$

where $|*|$ is the amount of unit in the set $*$. The value of $NAP(DIS_k)$ is between 0 and 1. The larger the value, the more significant the role of the source DIS_k in the classification task under attribute a. To better understand the calculation process of neighborhood approximate accuracy, we provide a relevant example in Example 3.2.

Example 3.2. Following Example 3.1, we firstly compute $\underline{NR_1^1(X_h)}$ of DIS_1 under attribute a_1. Based on the data in Table 1, we can compute $U/d = \{\{x_1, x_2, x_3\}, \{x_4, x_5, x_6\}, \{x_7, x_8, x_9\}\}$. At this point, $X_1 = \{x_1, x_2, x_3\}$, and $\underline{NR_1^1(X_1)} = \{x_2, x_3\}$. Similarly, we obtain $\underline{NR_1^1(X_2)} = \emptyset$, and $\underline{NR_1^1(X_3)} = \emptyset$. Then, we can compute $\underline{GNR_1^1(U)} = \underline{NR_1^1(X_1)} \cup \underline{NR_1^1(X_2)} \cup \underline{NR_1^1(X_3)} = \{x_2, x_3\}$. Then, neighborhood approximate accuracy $NAP^1(DIS_1) = \frac{|GNR_1^1(U)|}{|U|} = \frac{2}{9} \approx 0.22$.

3.3. Construction of graph

To fully consider the role of each source in the classification task and the linear correlation between sources, we construct a graph where the node set V consists of all sources, and the edge set E represents the connections between sources. The edge weights $w(e)$ are determined by comprehensively considering the contribution of both sources to the classification task and their linear relationship. Thus, we can define the values of the elements in the adjacency matrix W^a of the graph. For each W_{ij}^a in adjacency matrix W^a, we have

$$W_{ij}^a = \alpha N_{ij}^a + (1-\alpha) R_{ij}^a, \tag{10}$$

where i and j separately mean i-th and j-th DIS in MSDIS, α is a parameter set between 0 and 1. The value of W_{ij}^a ranges from 0 to 1. The higher W_{ij}^a is, the better the pair of DIS_i and DIS_j is. N_{ij}^a represents the role of i-th and j-th DIS in classification under attribute a. It can be calculated as follows:

$$N_{ij}^a = \max(NAP^a(DIS_i), NAP^a(DIS_j)), \tag{11}$$

where DIS_i and DIS_j separately mean i-th and j-th DIS in MSDIS. N_{ij}^a ranges from 0 to 1. The pair of DIS_i and DIS_j become more efficient for the classification task when N_{ij}^a is larger. R_{ij}^a represents the linear relationship between i-th and j-th DIS under attribute a. According to Equation (4), we define the formula for R_{ij}^a as follows:

$$R_{ij}^a = \left| \frac{n\sum x_p y_p - \sum x_p \sum y_p}{\sqrt{(n\sum x_p^2 - (\sum x_p)^2)}\sqrt{(n\sum y_p^2 - (\sum y_p)^2)}} \right|, \tag{12}$$

where x_p and y_p respectively represent the value of p-th sample $u_p \in U$ in DIS_i and DIS_j under attribute a, n represents the number of sample in MSDIS, $|*|$ is the absolute value of $*$. N_{ij}^a ranges from 0 to 1. When N_{ij}^a becomes larger, there is more linear correlation between DIS_i and DIS_j. We constructed a graph using the above method. To better understand the process of graph construction, we provide a numerical example in Example 3.3.

Example 3.3. In this example, we calculate the element W_{12}^1 in the adjacency matrix W^1 for attribute a_1. Similar to the calculation of $NAP^1(DIS_1)$ in Example 3.2, we also obtain $NAP^1(DIS_2) \approx 0.22$. Thus, $N_{12}^1 = \max(NAP^1(DIS_1), NAP^1(DIS_2)) = 0.22$. At the same time, we compute $R_{12}^1 \approx 0.9743$ with Equation (12). When the parameter $\alpha = 0.5$, we get $W_{12}^1 = 0.5 \times N_{12}^1 + 0.5 \times R_{12}^1 \approx 0.5983$. Using the same method, we obtain the graph adjacency matrix W^1 for attribute a_1 as follows:

$$W^1 = \begin{bmatrix} 0.6111 & 0.5983 & 0.5856 \\ 0.5983 & 0.6111 & 0.5791 \\ 0.5856 & 0.5791 & 0.6111 \end{bmatrix}.$$

Next, we will use a specific approach to select the optimal information source based on this graph as the final data for attribute a in DIS.

3.4. Source choosing process

In this subsection, we will introduce a method for selecting the optimal information source under feature a using the adjacency matrix W^a. For the convenience of formula representation, we use W to denote the adjacency matrix W^a under attribute a in this chapter. According the definition of path in subsection 2.4, given a path $\Gamma = v_1 v_2 \dots v_t$ in the graph G, the overall weight of path Γ is defined as follows:

$$\Omega_\Gamma = \prod_{q=1}^{t} W(v_q, v_{q+1}). \tag{13}$$

If there are multiple paths of length t from vertex v_i to vertex v_j, where i and j separately mean i-th and j-th DIS in MSDIS, these paths form a set η_{ij}^t. We define the total weight of this path set between the two vertices as follows:

$$\Theta^t(i,j) = \sum_{\Gamma \in \eta_{ij}^t} \Omega_\Gamma. \tag{14}$$

Since we have already constructed the adjacency matrix W of the graph, we can transform Equation (13) and Equation (14) into operations on the adjacency matrix. Thus, the total weight of the path set η_{ij}^t is calculated as follows:

$$\Theta^t = (W)^t, \tag{15}$$

where the $(*)^t$ represents the tth power of the matrix $*$. The element in the i-th row and j-th column of the matrix Θ_{ij}^t represents the total weight of the path set η_{ij}^t. In the case where the path length is t, to evaluate the quality of a given node v_i, we have the preference value of DIS_i under the path length t as follows:

$$\varphi^t(i) = \sum_{j=1}^{l} \Theta^t(i,j), \tag{16}$$

where l represents the number of DIS in MSDIS, which is the number of nodes in the graph. If the φ^t (i) is larger, it indicates that the corresponding DIS of the node is better under the path length t. To consider all path lengths, we need to sum the $\varphi^t(i)$ obtained for all path lengths, which gives:

$$\varphi(i) = \sum_{t=1}^{\infty} \varphi^t(i). \tag{17}$$

The larger $\varphi(i)$ is, the better the DIS corresponding to the node. By calculating $\varphi(i)$ for all DIS, we can select the information source with the highest value as the final result under attribute a. The step of Equation (16) and Equation (17) can also be represented using matrix operations as follows:

$$\vec{\varphi} = \sum_{t=1}^{\infty} (\Theta^t \vec{\beta}) = (\sum_{t=1}^{\infty} (W)^t)\vec{\beta}, \tag{18}$$

where $\vec{\beta}$ represents a unit column vector with l dimensions. l is the number of DIS in MSDIS. The i-th element in the vector $\vec{\varphi}$ represents the preference value of DIS_i. Since Equation (18) involves an infinite series, its convergence cannot be guaranteed, and the computational complexity is also high. Therefore, we need to transform Equation (18) to guarantee its convergence and make it easier to implement. Thus, we introduce Proposition 4.

Proposition 4. *For an adjacency matrix W of a graph, if the constant c satisfies $0 < c < \frac{1}{\rho(W)}$, we can use $\sum_{t=1}^{\infty}(cW)^t$ to approximate the calculation of the infinite series $\sum_{t=1}^{\infty}(W)^t$. $\rho(W)$ represents the spectral radius of the matrix W. It can be calculated by $\rho(W) = \max_{i=1,2,\ldots,l} |\lambda_i|$, where λ_i is the i-th eigenvalue of the matrix W.*

Proof 4. First, we discuss the convergence of Equation (18). According to $\sum_{t=0}^{\infty} Q^t$ converges $\iff \rho(Q) < 1$, for Equation (18) to converge, the spectral radius of the matrix W must be less than one. For any two matrices A and B, their spectral radius satisfy the following property $\rho(A)\rho(B) \geq \rho(AB)$. If matrix A is given as cE, where c is a constant and E is the identity matrix, then we have $\rho(cE)\rho(B) \geq \rho(cEB)$. According to the definition of spectral radius, we obtain $\rho(cE) = c$. Using matrix multiplication properties, we further derive $c\rho(B) \geq \rho(cB)$. If we need to ensure $\rho(cW) < 1$, it is sufficient to guarantee that $c\rho(W) < 1$, which simplifies to $c < \frac{1}{\rho(W)}$. Thus, by multiplying the matrix W by a constant $0 < c < \frac{1}{\rho(W)}$, we can ensure the convergence of Equation (18). Since multiplying a matrix by a constant does not alter the relative magnitude of its elements, this operation will not affect the final selection results. Next, we discuss the issue of simplifying the computation of Equation (18). If $\sum_{t=0}^{\infty} Q^t$ converges, then we have $\sum_{t=0}^{\infty} Q^t = (E-Q)^{-1}$. Since the formula involves summing from $t = 1$ instead of $t = 0$, we have the following formula $\sum_{t=1}^{\infty}(cW)^t = (E - cW)^{-1} - (cW)^0$. Since $(cW)^0 = E$, then we have $\sum_{t=1}^{\infty}(cW)^t = (E - cW)^{-1} - E$.

Thus, we have resolved the issues of Equation (18)'s convergence and computational complexity through this proposition. The final formula for calculating the $\tilde{\vec{\varphi}}$ becomes:

$$\tilde{\vec{\varphi}} = (\sum_{t=1}^{\infty}(cW)^t)\vec{\beta} = ((E - cW)^{-1} - E)\vec{\beta}, \tag{19}$$

where the constant $c = \frac{0.9}{\rho(W)}$. After obtaining the $\tilde{\vec{\varphi}}$, we select the DIS corresponding to the element with the highest value in the $\tilde{\vec{\varphi}}$. This gives us the final DIS under attribute a. Considering the large number of variables used in this subsection, we have organized all the variables into Table 2 for easier reference by the reader. To help readers understand the theory presented in this section, we provided a small illustrative example in Example 3.4.

Example 3.4. In this example, we continue from the previously computed graph adjacency matrix. Based on the preference value vector calculation method introduced in this section, we compute the preference value vector and select the optimal information source under a_1. According to the W^1 obtained in Example 3.3, we can obtain $\tilde{\vec{\varphi}}$ under a_1 with Equation (19) as follows:

Table 2
Table of variable descriptions in Section 3.4.

Variable	Description
Ω_Γ	the overall weight of path
$\Theta^t(i,j)$	the total weight of the path set η_{ij}^t
Θ^t	the matrix with $\Theta^t(i,j)$ as its (i, j)-th element
$\varphi^t(i)$	the preference value of DIS_i under the path length t
$\varphi(i)$	the preference value of DIS_i
$\vec{\varphi}$	the preference vector with $\varphi(i)$ as its i-th element
$\tilde{\vec{\varphi}}$	the estimated vector of $\vec{\varphi}$

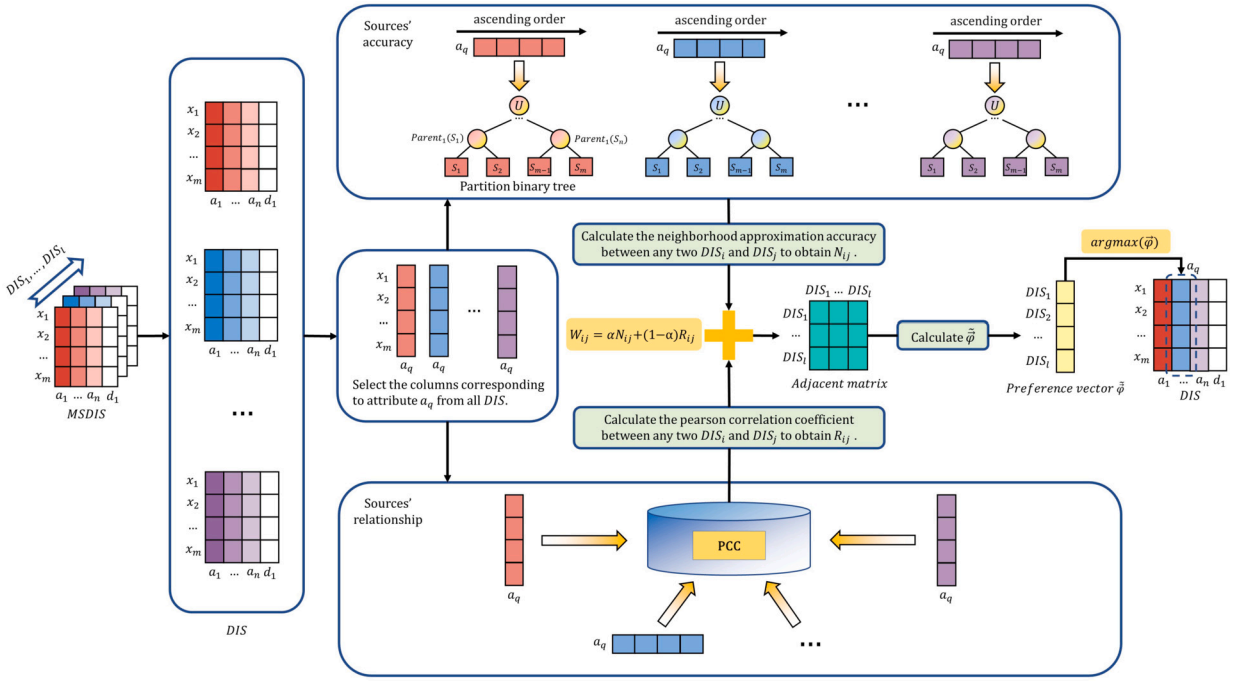

Fig. 2. The schematic diagram of the graph based information fusion method using GRNRS.

$$\tilde{\vec{\varphi}} = \begin{bmatrix} 9.0434 \\ 9.0104 \\ 8.9456 \end{bmatrix}.$$

Based on the calculated $\tilde{\vec{\varphi}}$ values, DIS_1 has the highest corresponding value. Therefore, we select the a_1 attribute column of DIS_1 as the a_1 attribute column in the final result.

3.5. The algorithm of the information fusion method

In this subsection, we summarize all the methods discussed above and propose an information fusion algorithm. The summary of each subsection in this section is listed below:

- In subsection 3.1, we introduced a method for constructing a partition binary tree using sorted samples and proposed how to obtain the neighborhood set from the constructed partition binary tree.
- In subsection 3.2, we explained how to use the obtained neighborhood set to calculate the neighborhood approximate accuracy for any DIS under attribute a.
- In subsection 3.3, we proposed a method for constructing a graph using the neighborhood approximate accuracy and the PCC.
- In subsection 3.4, we introduced a method to obtain the preference value for each DIS under attribute a using the constructed graph. Finally, we select the DIS with the highest preference value as the final result under attribute a.

To better explain the main process of the algorithm, Fig. 2 is the schematic diagram of the algorithm. Next, we propose the information fusion algorithm in Algorithm 2. The time complexity of steps 2-7 in Algorithm 2 is $O(l \cdot m^2)$, where l is the number of information sources and m is the number of samples. The time complexity of steps 8-14 is $O(l^2 \cdot m)$. The time complexity of steps 15-18 is $O(l^3)$. Thus, the overall time complexity of the algorithm is $O(n \cdot (l \cdot m^2 + l^2 \cdot m + l^3))$, where m is the number of samples, n is the number of attributes, and l is the number of information sources.

Algorithm 2: Information fusion algorithm based on GRNRS and graph theory.

Input: $MSDIS = \{DIS_k | DIS_k = (U, A, d, f_A^k, V_A^k, f_d^k, V_d^k), k = 1, 2, ..., l\}$
Output: $DIS_{out} = (U, A, d, f_A, V_A, f_d, V_d)$
for *each* $a \in A$ **do**
  for *each* DIS_k *in* $MSDIS$ **do**
    Sort the sample set U in ascending order to obtain U_{sorted}.
    Construct a partition binary tree T_k^a using U_{sorted}.
    Obtain the neighborhood set S through Algorithm 1.
    Calculate $NAP^a(DIS_k)$ through Equation (7), Equation (8) and Equation (9).
  end
  for *each* DIS_i *in* $MSDIS$ **do**
    for *each* DIS_j *in* $MSDIS$ **do**
      $N_{ij}^a \leftarrow \max(NAP^a(DIS_i), NAP^a(DIS_j))$
      Calculate R_{ij}^a through Equation (12).
      $W_{ij}^a \leftarrow \alpha N_{ij}^a + (1-\alpha) R_{ij}^a$
    end
  end
  $c \leftarrow \frac{0.9}{\rho(W^a)}$
  $\tilde{\vec{\varphi}} \leftarrow ((E - cW^a)^{-1} - E)\vec{\beta}$
  $p \leftarrow argmax(\tilde{\vec{\varphi}})$
  Set the attribute a column in DIS_p as the attribute a column in DIS_{out}.
end
return DIS_{out}

Table 3
The description of data sets.

No.	Data sets	Abbreviation	Samples	Attributes	Classes
1	Abalone	Abalone	4177	8	3
2	Avila	Avila	20867	10	12
3	Ecoli	Ecoli	336	7	8
4	Musk	Musk	476	168	2
5	Pendigits	Pendigits	10992	16	10
6	QSAR	QSAR	1055	41	2
7	Shill Bidding Dataset	SBD	6321	10	2
8	South German Credit	SGC	1000	21	2
9	Toxicity	Toxicity	171	1203	2
10	Waveform	Waveform	5000	21	3
11	Wholesale Customers Data	WCD	440	8	3
12	Wine	Wine	178	13	3

Table 4
Operating ambient.

Name	Model	Parameter
CPU	12th Gen Intel(R) Core(TM) i7-12700H	2.30 GHz
Platform	Python	3.9
System	Windows 11	64 bit
Memory	DDR5	16 GB; 4800 MHz
Hard Disk	Micron MTFDKBA512TFH	477 GB

4. Experiments and results

In this section, we analyze the impact of various parameters of the proposed method on the final performance through experimental analysis. Additionally, we evaluate the effectiveness and time efficiency of the proposed method by comparing it with other methods. In this experiment, all the data we use comes from UCI. The number of samples, attributes, and categories for each dataset will be presented in Table 3. The experiment is conducted on a personal computer, and its configuration will be shown in Table 4. Since the raw data does not meet the conditions for multi-source information, we need to preprocess the original data.

The preprocessing steps for the raw data are as follows. Given a $DIS = (U, A, d, f_A, V_A, f_d, V_d)$, we first apply normalization to ensure that all values in DIS fall within the range of 0 to 1. Next, we add the source data DIS to $MSDIS$. Then, we generate m random numbers $r = \{r_1, r_2, \ldots, r_m\}$, which follow a normal distribution $N(0, 0.1)$, where m represents the total number of samples. The value of sample x_i for attribute a in DIS_k is modified as $f_A^k(x_i, a) = f_A(x_i, a) + r_i, x_i \in u, a \in A$. Next, we need to assign values of 1 to elements greater than 1 and values of 0 to elements less than 0, in order to ensure that all values lie within the range of 0 to 1. Meanwhile, the decision attribute value of sample x_i remains unchanged as $f_d = f_d(x_i, d), x_i \in U$. As a result, we obtain $MSDIS = \{DIS_k | DIS_k = (U, A, d, f_A^k, V_A^k, f_d^k, V_d^k), k = 1, 2, ..., l\}$.

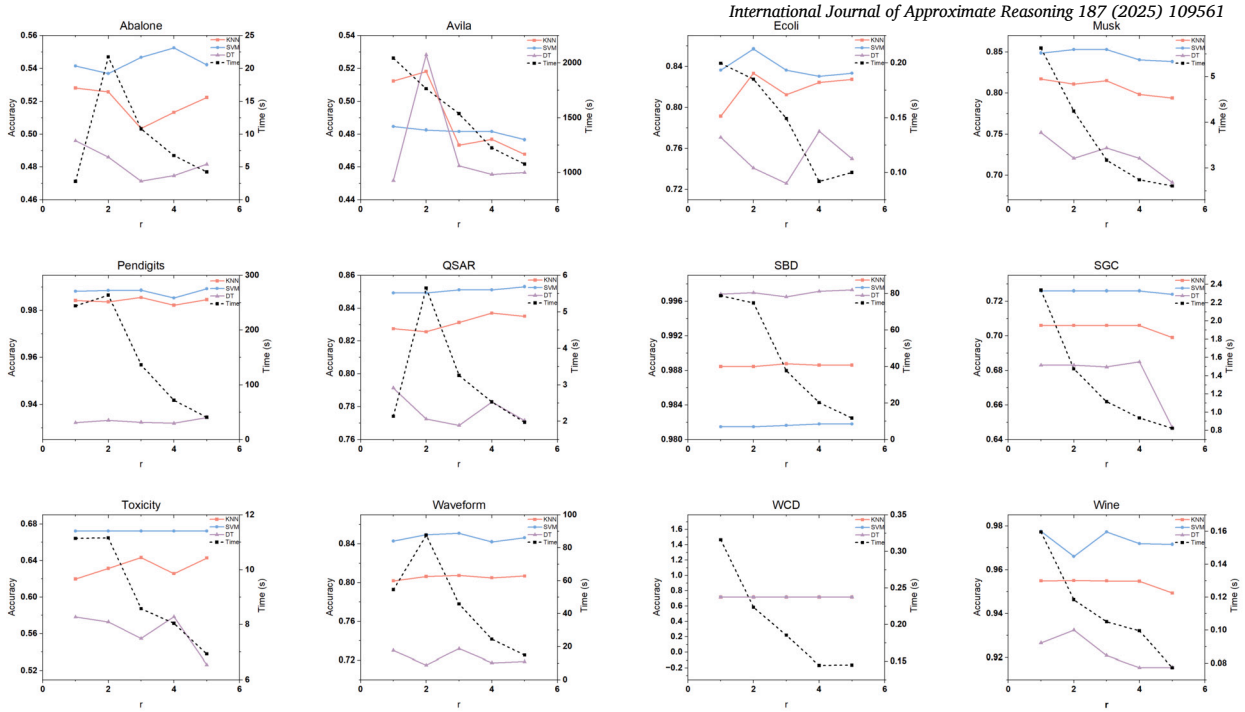

Fig. 3. The chart of the relationship between classification accuracy and runtime corresponding to different values of the parameter r in the algorithm.

4.1. Experimental designs

In this subsection, we will introduce the experimental design used to examine the impact of parameters in the proposed method on algorithm performance, as well as the experimental design for evaluating the efficiency and effectiveness of the proposed method.

4.1.1. The impact of parameters on the final performance of the algorithm

Our proposed method includes two parameters: one is the neighborhood radius r used in generating the neighborhood set, and the other is the weight parameter α used in constructing the adjacency matrix of the weighted undirected graph. We first examine the effect of parameter r on the final results of the algorithm.

To examine the impact of parameter r on the algorithm's results, we need to fix the value of parameter α. To equally consider both the linear relationship between information sources and their influence on classification, we set $\alpha = 0.5$. Since the height of the constructed partition binary tree is related to the number of samples in the dataset, as stated in Proposition 2, and according to its Definition 4, the value of neighborhood radius r must be smaller than the depth of the partition binary tree. Therefore, considering the dataset with the smallest number of samples, we set the range of r from 1 to 5 with a step size of 1.

To evaluate the information fusion results of the algorithm, we use three classifiers: K Nearest Neighbors (KNN), Support Vector Machine (SVM), and Decision Tree (DT) to classify the fused information and compute their classification accuracy. A higher classification accuracy indicates better information fusion performance of the algorithm. During model training, we adopt a 5-fold cross-validation strategy, and the final classification accuracy is obtained by averaging the accuracy across the five folds. At the same time, we record the computation time required for different values of parameter r, which helps analyze the impact of r on computational efficiency. The results of this experiment are shown in Fig. 3.

Next, we evaluate the impact of parameter α on the algorithm's performance. Based on the experimental results for parameter r, where the best performance was achieved at $r = 2$, we fix $r = 2$ in this experiment. Since α ranges from 0 to 1, we set its values from 0.1 to 0.9 with a step size of 0.1. Similar to the evaluation of r, we use KNN, SVM, and DT classifiers to classify the data and compute their classification accuracy. The experimental results are presented in Fig. 4.

4.1.2. The effectiveness of our algorithm comparing with other algorithms

In this subsection, we compare our proposed algorithm with other algorithms to gain a deeper understanding of its effectiveness. We will firstly introduce the comparison algorithms.

• Maximization Algorithm: This algorithm selects the maximum value from the multi-source data as the final fusion result. For the input $MSDIS = \{DIS_k | DIS_k = (U, A, d, f_A^k, V_A^k, f_d^k, V_d^k), k = 1, 2, ..., l\}$, the value of sample x under attribute a in the output $DIS = (U, A, d, f_A, V_A, f_d, V_d)$ can be expressed as $f_A(x, a) = \max_{k=1,2,...,l} f_A^k(x, a)$. The value of sample x under decision attribute d is the same as that in MSDIS.

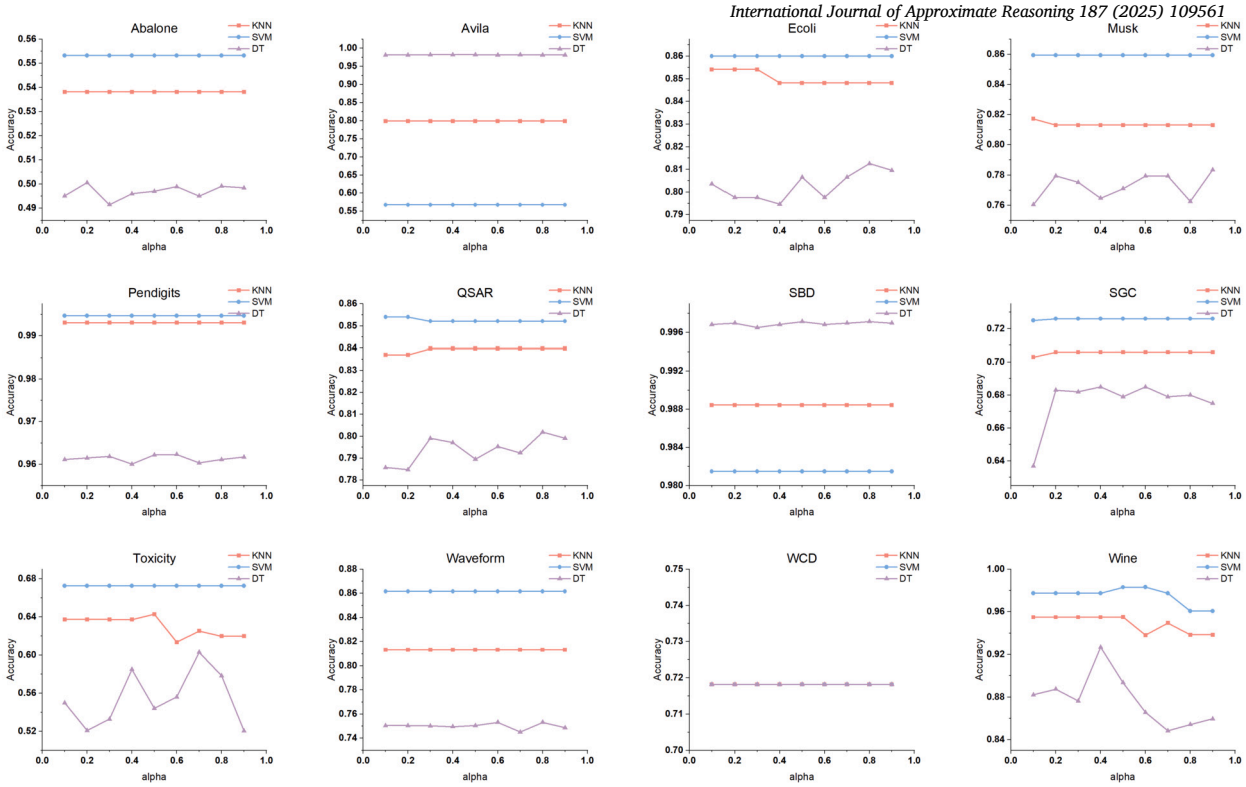

Fig. 4. The chart of the relationship between classification accuracy corresponding to different values of the parameter α in the algorithm.

• Minimization Algorithm: This algorithm determines the final fusion result by selecting the smallest value from the multi-source data. Given the input $MSDIS = \{DIS_k | DIS_k = (U, A, d, f_A^k, V_A^k, f_d^k, V_d^k), k = 1, 2, ..., l\}$, the value of sample x under attribute a in the output $DIS = (U, A, d, f_A, V_A, f_d, V_d)$ is defined as $f_A(x, a) = \min_{k=1,2,...,l} f_A^k(x, a)$. The value of sample x under decision attribute d is the same as that in MSDIS.

• Average Algorithm: This algorithm calculates the final fusion result by taking the average of the corresponding multi-source data values. For the input $MSDIS = \{DIS_k | DIS_k = (U, A, d, f_A^k, V_A^k, f_d^k, V_d^k), k = 1, 2, ..., l\}$, the value of sample x under attribute a in the output $DIS = (U, A, d, f_A, V_A, f_d, V_d)$ can be expressed as $f_A(x, a) = \frac{\sum_{k=1,2,...,l} f_A^k(x,a)}{l}$. The value of sample x under decision attribute d is the same as that in MSDIS.

• DI [30]: This method requires building a dependency function that considers both the interval length and the number of data points within it. Then, a median point within the interval is chosen as a reference to facilitate the determination of the dependency interval.

• IE [40]: This algorithm firstly establishes a tolerance relation based on RST. Utilizing this relation, a new conditional entropy is introduced to assess the significance of sources in relation to attributes. This approach determines the fusion result by selecting the attribute value of the source that corresponds to the lowest conditional entropy.

• DSET [39]: This algorithm introduces the concept of a support matrix by converting the data matrix into a support matrix. To handle evidence conflicts, it incorporates an additional data source based on average values. Moreover, a hierarchical fusion approach is proposed for further integration.

• FIE [31]: This algorithm constructs a fuzzy dominance relation between any two sources based on RST and FST. Additionally, a new conditional entropy is introduced to evaluate the importance of sources based on this relationship. For each attribute, the method calculates the conditional entropy in each source individually and identifies the source with the lowest result.

To evaluate the effectiveness of the algorithms, we introduce two metrics: classification accuracy and Average Quality (AQ). The AQ metric is a comprehensive measure to evaluate the performance of classification models. AQ combines Precision, Recall and F1-score to provide a balanced assessment of a model's classification quality. It is defined as follows:

$$\text{AQ} = \frac{1}{3}(\text{Precision} + \text{Recall} + \text{F1-score}),$$

where precision measures the proportion of predicted positive samples that are actually positive, recall measures the proportion of actual positive samples that are correctly predicted, F1-score is the harmonic mean of Precision and Recall, reflecting the balance between them. These metrics were originally designed for binary classification problems. To make them applicable to multi-class classification, we redefined them accordingly. The weighted formulas are as follows:

Table 5
Comparison of the classification accuracy of various algorithms after using KNN for classification tasks.

	raw	max	min	mean	DI	IE	DSET	FIE	our method
abalone	0.5344	0.4958	0.5073	0.5121	0.4987	0.4865	**0.5344**	0.5016	**0.5344**
avila	0.7980	0.3689	0.4058	0.4774	0.4210	0.3550	**0.7980**	0.3745	**0.7980**
ecoli	0.8662	0.8364	0.8126	0.8482	0.8423	0.7888	0.8305	0.7680	**0.8662**
musk	0.8361	0.8277	0.8236	0.8403	0.8319	0.8298	**0.8340**	0.8298	0.8277
pendigits	0.9923	0.9783	0.9819	0.9889	0.9863	0.9748	**0.9923**	0.9769	**0.9923**
QSAR	0.8578	0.8180	0.8066	0.8408	0.8256	0.7592	**0.8578**	0.7924	**0.8578**
SBD	0.9894	0.9881	0.9847	0.9875	0.9883	0.9834	**0.9894**	0.9872	**0.9894**
SGC	0.6940	0.7050	0.6950	0.7010	0.6970	0.6960	**0.7160**	0.7080	**0.7160**
Toxicity	0.6497	0.6440	0.6266	0.6497	0.6380	0.6435	**0.6612**	0.6324	0.6321
waveform	0.8228	0.788	0.778	0.8048	0.8074	0.7436	**0.8228**	0.7578	**0.8228**
WCD	0.7182	0.6432	0.6977	0.6273	0.6409	0.6523	0.6455	0.6614	**0.7182**
wine	0.9549	0.9494	0.9552	0.9606	0.9719	0.9608	0.9608	0.9330	**0.9776**

Table 6
Comparison of the classification accuracy of various algorithms after using SVM for classification tasks.

	raw	max	min	mean	DI	IE	DSET	FIE	our method
abalone	0.5535	0.5348	0.5372	0.5487	0.5456	0.5336	**0.5535**	0.5406	**0.5535**
avila	0.5684	0.4117	0.4536	0.4898	0.4577	0.4109	**0.5684**	0.4281	**0.5684**
ecoli	0.8780	0.8574	0.8453	0.8691	0.8602	0.8037	0.8424	0.8156	**0.8781**
musk	0.8677	0.8572	0.8614	**0.8719**	0.8656	0.8551	0.8677	0.8425	**0.8719**
pendigits	0.9944	0.9805	0.9834	0.9913	0.9890	0.9787	**0.9944**	0.9826	**0.9944**
QSAR	0.8569	0.8303	0.8161	0.8550	0.8408	0.7991	**0.8569**	0.8057	**0.8569**
SBD	0.9820	**0.9842**	0.98276	0.98212	0.9832	0.9813	0.9820	0.9820	0.9820
SGC	0.7250	0.7220	**0.7260**	0.7230	**0.7260**	**0.7260**	0.7250	**0.7260**	0.7250
Toxicity	0.6726	0.6726	0.6726	0.6726	0.6726	0.6726	0.6726	0.6726	0.6726
waveform	0.8604	0.8374	0.8286	0.855	0.8468	0.7932	**0.8604**	0.7972	**0.8604**
WCD	0.7182	0.7182	0.7182	0.7182	0.7182	0.7182	0.7182	0.7182	0.7182
wine	0.9775	0.9775	0.9775	0.9830	**0.9889**	0.9887	0.9832	0.9778	0.9832

Table 7
Comparison of the classification accuracy of various algorithms after using DT for classification tasks.

	raw	max	min	mean	DI	IE	DSET	FIE	our method
abalone	0.5028	0.4625	0.4647	0.4774	0.4659	0.4575	0.4927	0.4520	**0.5028**
avila	0.9864	0.2694	0.3050	0.3777	0.3252	0.2587	0.9856	0.2897	**0.9864**
ecoli	0.8007	0.7261	0.7648	0.7678	0.7739	0.7470	0.7379	0.6965	**0.8007**
musk	0.7920	0.6618	0.7417	0.7163	0.7311	0.7333	0.7857	0.6618	**0.7899**
pendigits	0.9597	0.9240	0.9214	0.9509	0.9347	0.9185	**0.9608**	0.9253	0.9597
QSAR	0.8095	0.7422	0.7251	0.7621	0.7318	0.729858	**0.8123**	0.7023	0.8095
SBD	0.9976	0.994	0.9935	0.9949	0.9948	0.9867	0.9975	0.9948	**0.9976**
SGC	0.6830	0.6150	0.6460	0.6160	0.6420	0.6040	**0.6830**	0.6310	**0.6830**
Toxicity	0.5321	0.5672	0.5617	**0.6086**	0.5555	0.5269	0.4914	0.6084	0.5909
waveform	0.7504	0.7174	0.7094	0.7474	0.7308	0.6628	0.7496	0.6518	**0.7504**
WCD	0.7182	0.5409	0.6614	0.5523	0.5364	0.5477	0.5455	0.5682	**0.7182**
wine	0.8819	0.9100	0.9441	0.9329	**0.9443**	0.9100	0.9103	0.8373	0.8706

$$\text{Precision} = \sum_{i=1}^{K} w_i \cdot P_i, \quad \text{Recall} = \sum_{i=1}^{K} w_i \cdot R_i, \quad \text{F1} = \sum_{i=1}^{K} w_i \cdot F1_i,$$

where K is the number of classes, w_i represents the proportion of samples in class i, and P_i, R_i, and $F1_i$ denote the Precision, Recall, and F1-score of class i, respectively. The resulting AQ value ranges from $[0, 1]$, where a higher value indicates better overall model performance in terms of accuracy, coverage, and stability. For each algorithm, we use KNN, SVM, and DT classifiers to classify the fused data and assess the results using the aforementioned evaluation metrics. We adopt a 5-fold cross-validation training strategy, where the final evaluation metrics are obtained by averaging the results from the 5 folds. The results are presented in Table 5, Table 6 and Table 7. Meanwhile, we performed Friedman and Nemenyi tests on the comparison results under the classification accuracy and AQ metrics. The specific evaluation method is as follows: For the classification accuracy metric, we calculate the average classification accuracy of each algorithm on every dataset by taking the mean of the results obtained from three classifiers (KNN, SVM, and DT). The Friedman and Nemenyi tests are then conducted using the average classification accuracies of all algorithms across all datasets. Similarly, the Friedman and Nemenyi tests can be conducted based on the AQ metric in the same manner. The results of Friedman test are presented in subsubsection 4.2.2. Fig. 5 and Fig. 6 respectively show the Critical Difference diagrams of the Nemenyi test based on the classification accuracy and AQ metrics.

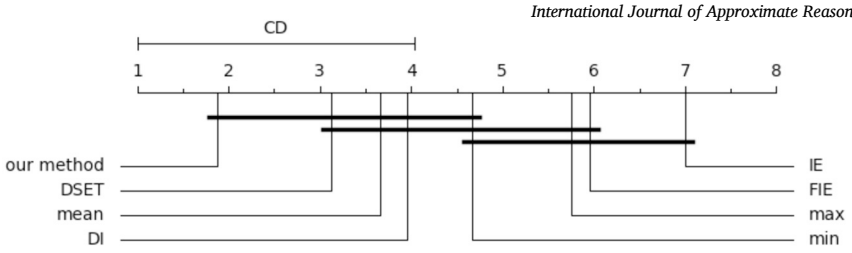

Fig. 5. The critical difference diagrams of the Nemenyi test based on the classification accuracy.

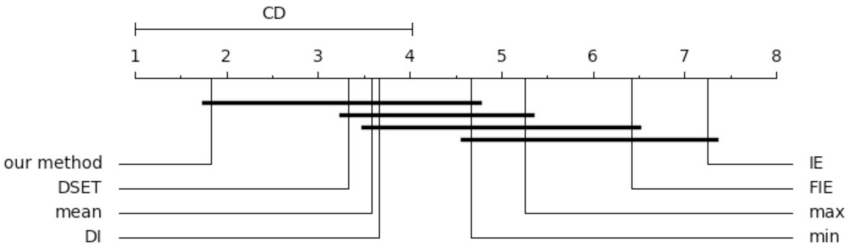

Fig. 6. The critical difference diagrams of the Nemenyi test based on the AQ.

Table 8
Comparison of the AQ of various algorithms after using KNN for classification tasks.

	raw	max	min	mean	DI	IE	DSET	FIE	our method
abalone	0.5355	0.4975	0.5076	0.5130	0.4995	0.4879	**0.5355**	0.5026	**0.5355**
avila-ts	0.7967	0.3255	0.3672	0.4516	0.3903	0.3050	**0.7967**	0.3124	**0.7967**
ecoli	0.8623	0.8336	0.8096	0.8455	0.8392	0.7815	0.8297	0.7622	**0.8623**
musk	0.8426	0.8344	0.8322	**0.8463**	0.8388	0.8363	0.8408	0.8370	0.8343
pendigits	0.9923	0.9783	0.9820	0.9889	0.9863	0.9749	**0.9923**	0.9734	**0.9923**
QSAR	0.8585	0.8189	0.8084	0.8414	0.8268	0.7577	**0.8585**	0.7919	**0.8585**
SBD	0.9895	0.9882	0.9846	0.9876	0.9884	0.9834	**0.9895**	0.9872	**0.9895**
SGC	0.6933	0.6971	0.6881	0.6927	0.6904	0.6855	0.6881	**0.7023**	0.6881
Toxicity	0.6339	0.6319	0.6029	0.6393	0.6217	0.6274	**0.6404**	0.6072	0.6154
waveform	0.8226	0.7876	0.7776	0.8046	0.8074	0.7432	**0.8226**	0.7574	**0.8226**
WCD	0.6115	0.6137	**0.6148**	0.5947	0.6037	0.5991	0.5947	0.5941	0.6115
wine	0.9559	0.9508	0.9567	0.9614	0.9725	0.9622	0.9613	0.9360	**0.9781**

Table 9
Comparison of the AQ of various algorithms after using SVM for classification tasks.

	raw	max	min	mean	DI	IE	DSET	FIE	our method
abalone	0.5417	0.5208	0.5265	0.5372	0.5349	0.5237	**0.5417**	0.5240	**0.5417**
avila-ts	0.5595	0.3222	0.3965	0.4316	0.4007	0.2810	**0.5595**	0.2805	**0.5595**
ecoli	0.8728	0.8515	0.8400	0.8655	0.8553	0.7994	0.8412	0.8086	**0.8728**
musk	0.8690	0.8581	0.8626	0.8731	0.8667	0.8558	0.8692	0.8433	**0.8731**
pendigits	0.9944	0.9806	0.9835	0.9913	0.9890	0.9788	**0.9944**	0.9768	**0.9944**
QSAR	0.8560	0.8285	0.8139	0.8538	0.8390	0.7976	**0.8560**	0.8029	**0.8560**
SBD	0.9825	0.9844	0.9830	0.9826	0.9835	0.9817	**0.9825**	0.9822	**0.9825**
SGC	0.7053	0.6996	**0.7081**	0.7019	0.7061	0.7043	0.7053	0.7053	0.7053
Toxicity	0.5553	0.5553	0.5553	0.5553	0.5553	0.5553	0.5553	0.5553	0.5553
waveform	0.8603	0.8374	0.8284	0.8549	0.8467	0.7931	**0.8603**	0.7970	**0.8603**
WCD	0.6115	0.6115	0.6115	0.6115	0.6115	0.6115	0.6115	0.6115	0.6115
wine	0.9780	0.9780	0.9777	0.9834	**0.9892**	0.9890	0.9836	0.9779	0.9836

4.1.3. The time efficiency of our algorithm comparing with other algorithms

To evaluate the time efficiency of our proposed algorithm, we recorded its execution time and compared it with the execution times of the baseline algorithms. The comparison algorithms used are DI [30], IE [40], SDET [39], and FIE [31], as introduced in subsubsection 4.1.2. The runtime comparison of these five algorithms across different datasets is presented in Table 11.

Table 10
Comparison of the AQ of various algorithms after using DT for classification tasks.

	raw	max	min	mean	DI	IE	DSET	FIE	our method
abalone	0.5026	0.4620	0.4651	0.4777	0.4656	0.4570	0.4929	0.4530	**0.5026**
avila-ts	0.9864	0.2712	0.3074	0.3803	0.3271	0.2613	0.9856	0.2963	**0.9864**
ecoli	0.7999	0.7269	0.7648	0.7655	0.7721	0.7436	0.7318	0.6917	**0.7999**
musk	0.7940	0.6613	0.7416	0.7169	0.7325	0.7339	0.7871	0.6628	**0.7907**
pendigits	0.9598	0.9241	0.9215	0.9509	0.9348	0.9185	**0.9608**	0.9202	0.9598
QSAR	0.8096	0.7433	0.7237	0.7642	0.7357	0.7318	**0.8120**	0.7016	0.8096
SBD	0.9976	0.9939	0.9935	0.9949	0.9948	0.9867	0.9975	0.9948	**0.9976**
SGC	0.6824	0.6182	0.6476	0.6153	0.6456	0.6068	0.6823	0.6319	**0.6824**
Toxicity	0.5426	0.5655	0.5641	0.6078	0.5647	0.5219	0.4846	0.6125	**0.5830**
waveform	0.7505	0.7176	0.7095	0.7474	0.7310	0.6632	0.7496	0.6521	**0.7505**
WCD	0.6115	0.5500	0.5933	0.5558	0.5515	0.5424	0.5533	0.5672	**0.6115**
wine	0.8837	0.9122	**0.9462**	0.9338	0.9455	0.9124	0.9120	0.8390	0.8719

Table 11
Comparison of the time taken by various algorithms to perform information fusion tasks.

	DI	IE	DSET	FIE	our method
abalone	0.77	1981.24	1145.21	7435.85	23.20
avila	4.90	72945.65	55826.85	156573.69	767.27
ecoli	0.06	8.72	6.19	18.14	0.21
musk	1.84	288.54	321.42	833.41	4.46
pendigits	4.50	17370.54	27529.67	87019.20	257.71
QSAR	0.99	356.04	381.40	1041.96	5.49
SBD	1.35	3588.41	5528.96	12318.33	47.33
SGC	0.49	156.22	163.27	519.25	2.21
Toxicity	4.93	275.08	598.35	858.21	12.86
waveform	2.62	4683.70	7699.44	11541.55	66.48
WCD	0.07	12.42	11.67	40.38	0.25
wine	0.05	3.23	3.13	9.34	1.26

4.2. Experimental results

In this subsection, we analyze the experimental results to determine how different parameters in the algorithm affect the final outcomes, as well as the algorithm's time efficiency and effectiveness.

4.2.1. The analysis of parameters' impact on the final performance of the algorithm

First, we analyze the impact of parameter r on the final results through Fig. 3. For the SVM classifier, the classification accuracy shows relatively small fluctuations as r varies, indicating that linear classifiers like SVM are not particularly sensitive to changes in r. For the KNN classifier, its classification accuracy remains largely unaffected by the variation of r in most classification tasks. However, in datasets where accuracy fluctuates more significantly, KNN tends to achieve higher accuracy when r is set to 1 or 2. The DT classifier, on the other hand, is more sensitive to changes in r. In most datasets, its classification accuracy varies considerably with different r values. By observing all the accuracy curves for DT, it can be concluded that for most datasets, setting r to 1 or 2 results in higher classification accuracy. Overall, selecting r as 1 or 2 tends to yield better classification accuracy across classifiers. Meanwhile, by observing the runtime curves, it can be seen that in most cases, the algorithm runs faster as the value of r increases. In conclusion, if the goal is to reduce the algorithm's runtime, a larger r value should be chosen. On the other hand, if the objective is to improve classification accuracy, a smaller r value may be more appropriate.

Next, we analyze the impact of the parameter α on classification accuracy using Fig. 4. By observing the curves for KNN and SVM, it can be seen that these two classifiers are not particularly sensitive to changes in α. In datasets with a large number of samples and few attributes, their classification accuracy remains stable. However, when faced with datasets that have few samples and many attributes, their classification accuracy fluctuates significantly, setting α around 0.5 tends to result in better classification performance. In contrast, the DT (Decision Tree) classifier is more sensitive to changes in α. Overall, it also tends to achieve higher classification accuracy when α is around 0.5. Therefore, setting α to 0.5 is generally a suitable choice for most situations.

4.2.2. The analysis of our algorithm's effectiveness comparing with other algorithms

According to Table 5, Table 6, Table 7, Table 8, Table 9 and Table 10, our algorithm performs well compared to other information fusion algorithms across all three classifiers. In most datasets, it achieves a noticeable improvement in classification accuracy and AQ, particularly when handling datasets with a large number of samples, such as Avila, Pendigits, and SBD However, in datasets with a large number of attributes, such as Musk and Toxicity, the algorithm does not perform as well. Compared with other algorithms, our method demonstrates greater advantages when using DT as the classification model. Moreover, compared with some RST-based algorithms such as IE and FIE, our proposed algorithm improves the classification accuracy of the information fusion results. This

may be due to the fundamental mechanism of the algorithm, which selects the best information source for each attribute individually without considering the relationships between attributes. As the number of attributes increases, the influence of any single attribute on the final classification performance diminishes. Consequently, evaluating each attribute in isolation becomes less effective in reflecting the overall impact of the information system on classification. Therefore, incorporating a mechanism that considers the information system as a whole during the fusion process would enable the algorithm to perform better when dealing with datasets that have many attributes but few samples. The p-values of the Friedman test under the classification accuracy and AQ metrics are 0.00000153 and 0.00000025, respectively. Using a significance level of 5%, we would reject the null hypothesis in favor of the alternative hypothesis: "The performance differences among the algorithms are statistically significant". According to Fig. 5 and Fig. 6, We can conclude that our algorithm demonstrates better efficiency compared to the other algorithms.

4.2.3. The analysis of our algorithm's time efficiency comparing with other algorithms

We analyze the time efficiency of our method based on the runtime comparison of the five algorithms presented in Table 11. Among all algorithms, the DI algorithm consistently has the shortest runtime; however, it also yields relatively low classification accuracy in information fusion tasks. Compared to the IE, DSET, and FIE algorithms, our method shows a significant improvement in runtime. This is especially evident in datasets with a large number of samples, such as Avila, Pendigits, and SBD, where our algorithm can save approximately 100 times more computation time compared to most other information fusion methods. Even in datasets with a large number of attributes, such as Musk and Toxicity, our algorithm demonstrates notably high computational efficiency compared to the majority of other fusion algorithms.

5. Conclusion and future work

In this section, we summarize the main content of this paper. First, we generate the neighborhood set using GRNRS. In this process, we propose an optimized algorithm for calculating the neighborhood set. Then, we introduce the concept of neighborhood approximate accuracy as a metric for evaluating the contribution of information sources to classification tasks under specific attributes. By combining PCC, we construct the adjacent matrix for information sources, where the elements represent the quality of the information source pairs. Next, we use a graph path algorithm to calculate the preference value of each information source. Finally, we select the information source corresponding to the maximum preference value as the final information source under the specific attribute a.

In the experimental section, we analyze the impact of parameter variations on the algorithm's results by comparing the classification accuracy of KNN, SVM, and DT classifiers for different values of parameters r and α. The experimental results indicate that as the value of r increases, the computation time decreases, while a smaller r leads to higher classification accuracy. When α is set to 0.5, a better classification accuracy is more likely to be achieved. Additionally, we compare our algorithm with several other algorithms and conduct the Friedman and Nemenyi test on the comparison results to assess our algorithm's effectiveness. The results show that our algorithm performs well across all three classifiers, especially when handling multi-sample data. However, its performance is less satisfactory when dealing with multi-attribute data. Finally, we analyze the time efficiency of our algorithm by comparing its execution time with that of various other algorithms. The results demonstrate that our algorithm significantly reduces the computation time compared to most information fusion algorithms, showcasing its high time efficiency.

Our algorithm still has considerable room for improvement and optimization. Firstly, the algorithm performed poorly in the experiments when handling multi-attribute datasets. As the possible reason proposed in subsubsection 4.2.2, future work can focus on incorporating a mechanism that holistically considers the entire information system during the fusion process to address this issue. Secondly, our algorithm currently only handles single-value multi-source information systems. In the future, modifications can be made to adapt the algorithm to handle other types of information systems.

CRediT authorship contribution statement

Yigao Li: Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Data curation. **Weihua Xu:** Supervision, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization.

Declaration of competing interest

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.

We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant 62376229, and Natural Science Foundation of Chongqing under Grant CSTB2023NSCQ-LZX0027.

Data availability

No data was used for the research described in the article.

References

[1] T.M. Al-shami, D. Ciucci, Subset neighborhood rough sets, Knowl.-Based Syst. 237 (2022) 107868.
[2] M. Boumediene, H. Zebiri, J. Dezert, Evidential data association based on Dezert–Smarandache theory, Int. J. Intell. Robot. Appl. 7 (2023) 91–102.
[3] K. Cai, W. Xu, An efficient multi-source information fusion approach for dynamic interval-valued data via fuzzy approximate conditional entropy, Int. J. Mach. Learn. Cybern. 15 (2024) 3619–3645.
[4] C. Chen, Q. Chen, G. Li, M. He, J. Dong, H. Yan, Z. Wang, Z. Duan, A novel multi-source data fusion method based on Bayesian inference for accurate estimation of chlorophyll-a concentration over eutrophic lakes, Environ. Model. Softw. 141 (2021) 105057.
[5] S. Cheng, B. Zhang, G. Zou, M. Huang, Z. Zhang, Friend recommendation in social networks based on multi-source information fusion, Int. J. Mach. Learn. Cybern. 10 (2019) 1003–1024.
[6] Y. Dong, X. Li, J. Dezert, M.O. Khyam, M. Noor-A-Rahim, S.S. Ge, Dezert-Smarandache theory-based fusion for human activity recognition in body sensor networks, IEEE Trans. Ind. Inform. 16 (2020) 7138–7149.
[7] A. Farasat, G. Gross, R. Nagi, A.G. Nikolaev, Social network analysis with data fusion, IEEE Trans. Comput. Soc. Syst. 3 (2016) 88–99.
[8] L. Fei, T. Li, W. Ding, Dempster–Shafer theory-based information fusion for natural disaster emergency management: a systematic literature review, Inf. Fusion 102585 (2024).
[9] L. Fei, T. Li, X. Liu, W. Ding, A novel multi-source information fusion method for emergency spatial resilience assessment based on Dempster-Shafer theory, Inf. Sci. 686 (2025) 121373.
[10] M. Hu, E.C. Tsang, Y. Guo, D. Chen, W. Xu, A novel approach to attribute reduction based on weighted neighborhood rough sets, Knowl.-Based Syst. 220 (2021) 106908.
[11] M. Li, F. Wang, X. Jia, W. Li, T. Li, G. Rui, Multi-source data fusion for economic data analysis, Neural Comput. Appl. 33 (2021) 4729–4739.
[12] R. Li, H. Chen, S. Liu, K. Wang, S. Liu, Z. Su, Outlier detection based on multisource information fusion in incomplete mixed data, Appl. Soft Comput. 165 (2024) 112104.
[13] S. Li, H. Xu, J. Xu, X. Li, Y. Wang, J. Zeng, J. Li, X. Li, Y. Li, W. Ai, Inconsistency elimination of multi-source information fusion in smart home using the Dempster-Shafer evidence theory, Inf. Process. Manag. 61 (2024) 103723.
[14] G. Lin, J. Liang, Y. Qian, An information fusion approach by combining multigranulation rough sets and evidence theory, Inf. Sci. 314 (2015) 184–199.
[15] K. Mao, Y. Wang, J. Ye, W. Zhou, Y. Lin, B. Fang, Belief structure-based Pythagorean fuzzy entropy and its application in multi-source information fusion, Appl. Soft Comput. 148 (2023) 110860.
[16] Z. Pawlak, Rough set theory and its applications to data analysis, Cybern. Syst. 29 (1998) 661–688.
[17] S. Qiu, H. Zhao, N. Jiang, Z. Wang, L. Liu, Y. An, H. Zhao, X. Miao, R. Liu, G. Fortino, Multi-sensor information fusion based on machine learning for real applications in human activity recognition: state-of-the-art and research challenges, Inf. Fusion 80 (2022) 241–265.
[18] B. Sang, L. Yang, W. Xu, H. Chen, T. Li, W. Li, VCOS: multi-scale information fusion to feature selection using fuzzy rough combination entropy, Inf. Fusion 117 (2025) 102901.
[19] L. Sun, T. Wang, W. Ding, J. Xu, Y. Lin, Feature selection using fisher score and multilabel neighborhood rough sets for multilabel classification, Inf. Sci. 578 (2021) 887–912.
[20] W.T. Sung, K.Y. Chang, Evidence-based multi-sensor information fusion for remote health care systems, Sens. Actuators A, Phys. 204 (2013) 1–19.
[21] X. Tao, J.D. Velasquez, Multi-source information fusion for smart health with artificial intelligence, 2022.
[22] B. Wang, C. Zhang, A.K. Sangaiah, M.J. Alenazi, S.A. AlQahtani, S.K. KS, Group consensus-driven energy consumption assessment using social network analysis and fuzzy information fusion, Int. J. Semantic Web Inf. Syst. 20 (2024) 1–32.
[23] J. Wang, W. Xu, W. Ding, Y. Qian, Multi-view fuzzy concept-cognitive learning with high-order information fusion of fuzzy attributes, IEEE Trans. Fuzzy Syst. (2024).
[24] N. Wang, E. Zhao, A new method for feature selection based on weighted k-nearest neighborhood rough set, Expert Syst. Appl. 238 (2024) 122324.
[25] Q. Wang, Y. Qian, X. Liang, Q. Guo, J. Liang, Local neighborhood rough set, Knowl.-Based Syst. 153 (2018) 53–64.
[26] W. Wei, J. Liang, Information fusion in rough set theory: an overview, Inf. Fusion 48 (2019) 107–118.
[27] S. Xia, C. Wang, G. Wang, X. Gao, W. Ding, J. Yu, Y. Zhai, Z. Chen, GBRS: a unified granular-ball learning model of Pawlak rough set and neighborhood rough set, IEEE Trans. Neural Netw. Learn. Syst. (2023).
[28] S. Xia, S. Wu, X. Chen, G. Wang, X. Gao, Q. Zhang, E. Giem, Z. Chen, GRRS: accurate and efficient neighborhood rough set for feature selection, IEEE Trans. Knowl. Data Eng. 35 (2022) 9281–9294.
[29] F. Xiao, Z. Cao, C.T. Lin, A complex weighted discounting multisource information fusion with its application in pattern classification, IEEE Trans. Knowl. Data Eng. 35 (2022) 7609–7623.
[30] W. Xu, Y. Lin, N. Wang, A novel multi-source information fusion method based on dependency interval, IEEE Trans. Emerg. Top. Comput. Intell. 8 (4) (2024) 3180–3194.
[31] W. Xu, Y. Pan, X. Chen, W. Ding, Y. Qian, A novel dynamic fusion approach using information entropy for interval-valued ordered datasets, IEEE Trans. Big Data 9 (2022) 845–859.
[32] V. Yaghoubi, L. Cheng, W. Van Paepegem, M. Kersemans, A novel multi-classifier information fusion based on Dempster–Shafer theory: application to vibration-based fault detection, Struct. Health Monit. 21 (2022) 596–612.
[33] L. Yang, X. Zhang, W. Xu, B. Sang, Multi-granulation rough sets and uncertainty measurement for multi-source fuzzy information system, Int. J. Fuzzy Syst. 21 (2019) 1919–1937.
[34] X. Yang, H. Chen, T. Li, C. Luo, A noise-aware fuzzy rough set approach for feature selection, Knowl.-Based Syst. 250 (2022) 109092.
[35] X. Yang, Z. Hua, L. Li, X. Huo, Z. Zhao, Multi-source information fusion-driven corn yield prediction using the random forest from the perspective of agricultural and forestry economic management, Sci. Rep. 14 (2024) 4052.
[36] Y. Yin, L. Zhang, W. Liao, H. Niu, F. Chen, A knowledge resources fusion method based on rough set theory for quality prediction, Comput. Ind. 108 (2019) 104–114.
[37] C. Zhang, D. Li, X. Kang, D. Song, A.K. Sangaiah, S. Broumi, Neutrosophic fusion of rough set theory: an overview, Comput. Ind. 115 (2020) 103117.
[38] P. Zhang, T. Li, G. Wang, C. Luo, H. Chen, J. Zhang, D. Wang, Z. Yu, Multi-source information fusion based on rough set theory: a review, Inf. Fusion 68 (2021) 85–117.
[39] Q. Zhang, P. Zhang, T. Li, Information fusion for large-scale multi-source data based on the Dempster-Shafer evidence theory, Inf. Fusion 115 (2025) 102754.
[40] X. Zhang, X. Chen, W. Xu, W. Ding, Dynamic information fusion in multi-source incomplete interval-valued information system with variation of information sources and attributes, Inf. Sci. 608 (2022) 1–27.
[41] X. Zhang, X. Shen, Graph-driven feature selection via granular-rectangular neighborhood rough sets for interval-valued data sets, Appl. Soft Comput. 170 (2025) 112716.

[42] K. Zhou, N. Lu, B. Jiang, Basic probability assignment using intuitive fuzzy cloud model for information fusion and its application in fault diagnosis, IEEE Trans. Instrum. Meas. 73 (2023) 1–13.
[43] C. Zhu, F. Xiao, A belief Rényi divergence for multi-source information fusion and its application in pattern recognition, Appl. Intell. 53 (2023) 8941–8958.